
AFIT/DS/ENG/99-06

# Image Fusion Using Autoassociative-Heteroassociative Neural Networks

*Claudia V. Kropas-Hughes, B.S., M.S.*

May 1999



DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/DS/ENG/99-06

# IMAGE FUSION USING AUTOASSOCIATIVE-HETEROASSOCIATIVE NEURAL NETWORKS

DISSERTATION

Presented to the Faculty of the Graduate School of Engineering
Of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

Claudia V. Kropas-Hughes, B.S., M.S.

May 1999

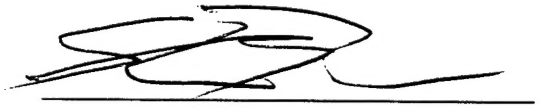
AFIT/DS/ENG/99-06

IMAGE FUSION USING

AUTOASSOCIATIVE-HETEROASSOCIATIVE NEURAL NETWORKS

Claudia V. Kropas-Hughes, B.S., M.S.

Approved:



5-25-99

Steven K. Rogers, Ph.D.
Chairman, Advisory Committee

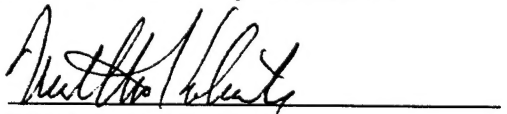
Date

Mark E. Oxley

19 Mar 99

Mark E. Oxley, Ph.D.
Member, Advisory Committee

Date



19 May 99.

Matthew Kabrisky, Ph.D.
Member, Advisory Committee

Date

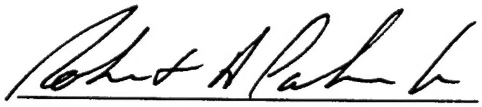
Peter S. Maybeck

19 May 99

Peter Maybeck, Ph.D.
Dean's Representative

Date

Accepted:



Robert A. Calico, Jr., Ph.D.
Dean, Graduate School of Engineering

*Acknowledgments*

*For my parents,*
*Virginia and George Kropas*

First, I wish to thank Dr. Steve Rogers for his guidance and encouragement throughout my program at AFIT. His patience and support have made it possible for me to reach this goal.

I am also grateful to Dr. Matthew Kabrisky, who shared his wisdom and enthusiasm and always found time to discuss my research.

I wish to thank Dr. Mark Oxley who spent many days with me, ensuring that I had a sound theoretical basis for my work.

Capt. Kelly Greene was always willing to share and commiserate; Capt. John Keller was a great sounding board, listener, and advisor; and Ms. Mary Jane McCormick kept my records and me on the straight and narrow. Many other faculty, staff, and fellow students assisted in this effort. For their support, I am grateful.

I wish to thank Dr. Steve LeClair and the management at the Air Force Research Laboratory for affording me the opportunity to participate in this program.

I am indebted to my family, Jeannie, Maryann, George, Lynne, and Gib, who encouraged and supported me.

Finally, I am extremely grateful to my husband, Michael Hughes, for his unfailing support and confidence in me. I truly appreciate the sacrifices he made to help me achieve this personal goal.

Claudia V. Kropas-Hughes

*Table of Contents*

*List of Figures*

*List of Tables*

*Abstract*

Images are easily recognized, classified, and segmented -- in short, analyzed -- by humans. The human brain/nervous system -- the biological "computer" -- performs rapid and accurate image processing. In the current research the concepts of the biological neural system provide the impetus for developing a computational means of fusing image data. Accomplishing this automatic image processing requires features be extracted from each image data set, and the information content fused. Biologically inspired computational models are examined for extracting features by transformations such as Fourier, Gabor, and wavelets, and for processing and fusing the information from multiple images through evaluation of autoassociative neural networks (AANNs). Features are obtainable through the use of human-visual-system models, however, AANNs are limited in accomplishing the desired data fusion. In-depth analysis of these networks demonstrate their functionality as data filters requiring careful feature selection to provide the desired image processing. Some features, when using perceptual space concepts, require AANNs to have the ability to process complex-valued inputs instead of only real-valued data. Human-visual-system concepts provide an alternative error function metric for training the networks based on the human standard of similarity instead of absolute numerical value. The most significant limitation of AANNs, for this data fusion application, is their inability to process multiple image data sources. The AANN concepts are extended to a new architecture -- the Autoassociative-Heteroassociative Neural Network (A-HNN) -- developed for predicting one sensor image from another by using input data values as desired target outputs. This new

architecture, using input data as desired outputs, provides better training performance and is an improvement over other neural network architectures. The effectiveness of the new architecture, for a classification application, is demonstrated using XOR two-class problem data. As an image fusion device, the A-HNN is demonstrated on medical Computed Axial Tomography (CT) and Magnetic Resonance (MR) image data using design concepts and principles from the human brain and nervous system to extract features.

# IMAGE FUSION USING
# AUTOASSOCIATIVE-HETEROASSOCIATIVE NEURAL NETWORKS

## *1 Introduction*

In many instances two sensors measure the same object, with each sensor providing a different perspective of the original object being examined. Often two experts are required -- one to analyze each sensor output. A computer algorithm for determining the same features present in the two sensors, in a manner similar to that used by the human analysts, would provide solutions to a number of problems, for example, in medical-image processing where Computed Axial Tomography (CT) scans and Magnetic Resonance (MR) Imaging scans are compared. Humans can quickly associate the same features in these two images, even when the gray-scale pixels indicate different properties, e.g., white pixels in CT images indicate high-density material such as bone and white pixels in MR images signify high water content such as blood.

The ability of the human biological computer to process image data accurately and efficiently provided the inspiration for use of the concepts and design principles of the neural system to process image data computationally. The human neural system has been studied extensively. One of the primary sensors of this system is the visual system. Models have been developed for mathematical simulation of the automatic processes performed by the human eye. "Seeing" and "understanding" of what has been received from the sensory inputs was the first step to be modeled. The concept of modeling from

the most basic brain unit, i.e., the neuron, has been implemented in computers as neural networks -- electronic circuits for mimicking the brain functions that process information (Hinton, 1993).

Examination of the human visual system reveals many properties of interest in extracting specific features from different sensor data. Since this system operates in a transform space (Baron, 1987; Field, 1990; Kabrisky, 1966; Maher, 1970; McLachlan, 1962), Fourier, Gabor, and wavelet transforms are potential means of reducing image data sets to those features that the human would use to correlate different images. However, extracting features is merely the first step. Once the features have been determined from each image, a means must be found of associating those features. Artificial neural networks can be used as models for the multi-layer processing that occurs in the brain for computationally associating different sensor data. The autoassociative neural network (AANN) is a specific architecture known for its use in image processing (Cottrell and others, 1989; DeMers and Cottrell, 1993; Hecht-Nielsen, 1990 and 1995; Kramer, 1991; Turk and Pentland, 1991). However, AANNs process only single-image data sets and real-valued data. The research documented in this dissertation has extended the applications of AANNs through:

- Their novel interpretation as filters to assist in further analysis of the network operations and feature-selection criteria
- Demonstration of their ability to process complex-valued data
- Proposal of an alternative error function for training image-processing neural networks

This dissertation documents the development of a new neural network architecture that can predict one sensor output from another sensor's input while providing a quantitative means for evaluating the robustness of the network; this new architecture is called the Autoassociative-Heteroassociative Neural Network (A-HNN). The new architecture is demonstrated on medical CT and MR image data. The features are extracted using concepts and design principles from the human visual system. The experimental results demonstrate that an understanding of the human biological computer can be used to provide a means of processing two different images of the same material.

### *1.1 Problem Statement*

This dissertation focuses on the development of a computational means of mimicking the human brain and nervous system to analyze two different image representations of the same material. Since the human neural system is a powerful, accurate, and efficient image processor and analyzer, mathematical models of this system have been used as the guide for relating and predicting one image data set from another.

As a specific example, Figure 1 shows two 48 x 48 gray-scale pixel images. The MR image on the left is of the back of a head, and the CT image on the right is of the back of the same head taken at the same location. The medical MR and CT images used in this study were provided by the Visible Human Project from the National Library of Medicine at the National Institutes of Health (Visible Human Project, email: vhp@nlm.hih.gov).

The image data were obtained from the same individual and at the same physical location. Therefore, the CT and MR images are registered to the same spatial location

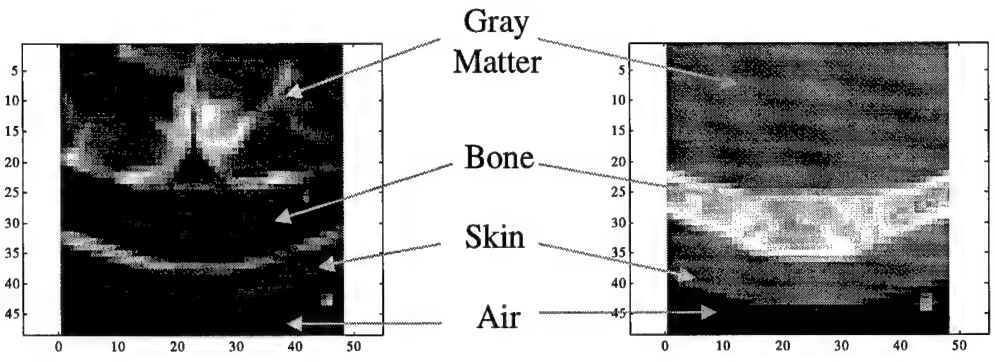



**Figure 1. MR Image, Left; CT Image, Right.**

and orientation; however, the resolution of the two methodologies is inherently different, being based on the sensor device that created the images. In this research effort the prediction of images in two dimensions was examined; since these images represent the same location, registration did not need to be performed. A brief description of the data-acquisition technique used for obtaining CT and MR images follows.

MR images are created by measuring the relaxation of the dipole potential of an element in an electric field, and MR scanners are tuned to specific elements. For medical images, the scanners are tuned for hydrogen (H), and MR images display the amount of H present in the material. In the case of humans, soft tissue and blood have the highest concentrations of H, making MR a water-content detector. The gray-scale values are interpreted as follows: black pixels represent no material; white pixels represent a high concentration of H; and the gray levels represent the various levels of H present in the material (Mansfield and Morris, 1982).

CT images are created by an x-ray technique. Multiple images are obtained from the same location in the material at different angular orientations. A single "slice" is reconstructed from the multiple through-body images using a technique called backprojection. The intensity of the x-ray source influences the interpretation of the

resulting image. For medical CT scanners low-energy x-rays are used, and the x-ray measurements are closely related to the density of the material being examined. In the CT image in Figure 1, white pixels represent high-density material, and black pixels represent air (Newton and others, 1981). The correlation between the two images in Figure 1 is high. A cursory look will reveal, even to non-experts, that these images are of the same material. However, the interpretation of each image must be based on the sensor modality (data-acquisition technique) from which it was generated.

This dissertation documents a means of analyzing and synthesizing or predicting an image from another representation of the same material using physiologically motivated concepts. Since the human brain and nervous system have incredibly powerful and efficient image-processing capabilities, human-visual-system concepts are examined as a guide for extracting features from different data sets. With the aid of the transformation properties of the human visual system, Fourier, Gabor and wavelet transforms are examined for their ability to extract pertinent features. Associating the features from the two different images is the next step; with inspiration from the human machine, neural networks are examined and a new architecture (A-HNN) applied for processing the features determined by the human visual system.

### *1.2 Outline of Dissertation*

The second chapter discusses previous research in the areas of data fusion and mathematical modeling of the biological neural system -- the brain and the nervous system. The human-visual-system concepts are specifically described as they relate to feature extraction and the current applications of artificial neural networks for image processing, with focus on the AANN.

The third chapter documents the new research on understanding and utilizing the concepts of visual processing for extracting features and correlating information from image data. The understanding of AANNs is extended to filtering data as a means of analyzing information, and the application of these networks is extended to a new architecture that incorporates the ability to associate two different images. This new architecture, the A-HNN, provides a means of predicting one sensor image from another and improves the training performance of the network.

The fourth chapter discusses the experimental results of this research effort. Feature extraction and the new A-HNN are demonstrated on sample medical images. The fifth chapter contains conclusions and outlines areas for future effort.

Appendix A proposes an alternative error function for training an image-processing neural network -- specifically, for processing image data. This metric, the Visual-Difference Predictor (VDP), provides a pixel-by-pixel comparison of two different images and is a "human-perceptual-space" evaluation of similarity between them. For images, similarity is subjectively determined and is unrelated to the absolute pixel values. Therefore, pixel values are not necessarily the most effective means of comparing image data. Training of image-processing neural networks driven by human perceptual standards of similar or "close enough to be the same" rather than absolute pixel values may be more effective. Appendix A describes the use of the VDP as an alternative metric for training an image-processing neural network to a human perceptual standard of similar.

Appendix B discusses the problems and considerations in processing complex-valued data in artificial neural networks. There are applications, image and non-image

data, in which information fusion is required and the data is complex-valued. Neural networks are constructed to process real-valued data. Special provisions and data pre-processing must be accomplished before a neural network can effectively manipulate the complex-valued numbers. Appendix B discusses options for processing complex-valued data through artificial neural networks and specifically AANNs.

Appendix C describes the trial and error process for artificial neural network architecture selection. The trial and error process necessitates a trade-off among many factors. Feature selection, network architecture -- number of layers and number of nodes per layer -- and training time are the key elements in developing a neural network. The overall process is an iterative one, with a discussion of the trade-offs described in Appendix C.

Appendix D extends the concepts of the A-HNN to classification data. The fundamental concept of the A-HNN is to train the network to a desired output, while using inputs as target data. The addition of the input data to the target vector, not only provides a means of determining the generalization robustness of the network, but also improves the training performance of the network. This concept is demonstrated in Appendix D.

# 2 *Background*

## *2.1 Introduction*

The focus of the research documented in this dissertation has been to determine a means of associating the same features in two image data sets and extending that association through the ability to predict one image data set from another. To provide an historical perspective, this chapter discusses past efforts on the problem of associating or correlating two sets of data -- often referred to as data fusion, a term used to indicate algorithms for associating or correlating the same features in two or more images from different sources. Many approaches to data fusion have been investigated, but most techniques require two steps -- extracting the features and associating the features (Bloch, 1996; Burt and Kolczynski, 1993; Chinzei and others, 1992; Grimson, 1995; Grimson and others, 1996; Lin and others, 1994; Murphy, 1996; Soltanian-Zadah and others, 1994; Undrill and others, 1992; Van den Elsen and others, 1995; Wang and others, 1996; Wilson, 1995). Accomplishments in data fusion are very application and problem specific, i.e., no typical or generic means of data fusion is relative to all applications or problems.

As described in Chapter 1, this dissertation details an investigation using image data sets. Processing of image data, as opposed to numerical data, involves issues that must be addressed in a different way. For instance, image data is typically large in volume, with much of the information of interest, or features, being depicted as regions instead of single data points. For the current application, the approach has been to examine the ways in which the human biological "computer" "sees" and "understands"

images. The human brain and nervous system have demonstrated extraordinary effectiveness in processing sensory information. The effectiveness of biological computation has inspired the study of human processes as a means of attaining similar capabilities using artificial devices based on the human neural systems (Tank and Hopfield, 1990).

Two areas must be investigated -- feature extraction and feature association. In the case of image data, human-visual-system capabilities serve as a guide for reducing the total image to the simple features of interest. Accomplishments in the modeling of human capabilities for processing data will be discussed as well as human-visual-system models which were studied for guidance in extracting pertinent image features from image data.

The human brain has the capacity to assimilate new information based on current inputs as well as to interpolate information based on past learning. This capability has inspired the concepts of artificial neural networks as devices for learning relationships. Artificial neural networks, designed to mimic the biological computations that humans perform, have been examined as a means of processing the features extracted. These networks have been used to develop the functional relationships within different image data sets, theoretically in a manner similar to that used by humans to associate the information.

In this chapter a specific neural network architecture, the autoassociative neural network (AANN), will be discussed. The AANN has been used extensively as an image-processing device for many years. The understanding and current applications of AANNs will be discussed and their capabilities and limitations presented. The purpose

of the current research has been to determine a means of analyzing and predicting image data. Since AANNs, as image processors, are designed for processing one image at a time, AANN concepts can be extended to heteroassociative neural networks (HANNs) and these concepts have been examined for application to this problem of predicting or synthesizing one image data set from another.

Past work in the area of data fusion will be discussed as well as the problem-specific means of extracting features for fusing data. The biological computation models that have been developed, inspired by the computational powers of the human nervous system, will also be summarized. Specifically, models of the human visual system will be discussed as a guide to feature extraction, and the concepts of artificial neural networks will be discussed as a means of analyzing and associating image feature data.

## *2.2 Data Fusion*

The term data fusion refers to the combining of information from multiple sources -- the ability to match the same features from different sources to provide more information than is available from a single image. A considerable amount of research has been performed on mapping data from multiple sources (Bloch, 1996; Burt and Kolczynski, 1993; Chinzei and others, 1992; Grimson, 1995; Grimson and others, 1996; Lin and others, 1994; Murphy, 1996; Soltanian-Zadah and others, 1994; Undrill and others, 1992; Van den Elsen and others, 1995; Wang and others, 1996; Wilson, 1995). For images, the primary emphasis has been on edge-detection algorithms for extracting features which are used for landmark correlation or matching edge patterns from different images.

Edge-detection algorithms are often used to generate edge features from different images that can be correlated. These features are then used in pattern-recognition algorithms to correlate images from different modalities by correlation of known landmarks, or atlas matching. In medical imaging, this technique has proven to be the most convenient means of mapping data because anatomical models are readily available for the human body. Many researchers delineate the edges of different features that represent specific anatomical structures. Correlating the edge feature extracted from each image allows registration of the separate images. These edge features are then used as the features for classification against templates or other *a priori* information (Chinzei and others, 1992; Grimson, 1995; Grimson and others, 1996; Lin and others, 1994; Soltanian-Zadah and others, 1994; Undrill and others, 1992; Van den Elsen and others, 1995; Wang and others, 1996).

Medical-imaging data fusion has very specific limitations. Medical data from various sources provide unique information; however, the source of the information is a factor that must be considered in interpretation of the data. Therefore, the context associated with the data must be preserved during data fusion. For medical imaging, fusion of data such as maximum values, weighted averages, and gradients from different modalities cannot be context-independent (Burt and Kolczynski, 1993; Wilson, 1995). The informational content must be taken into account because different modalities intrinsically provide different types of information (Bloch, 1996; Murphy, 1996). Thus, most of the research on mapping of medical data has been conducted through the use of standard pattern recognition against a template or anatomical model (Chinzei and others, 1992; Grimson, 1995; Grimson and others, 1996; Lin and others, 1994; Soltanian-Zadah

and others, 1994; Undrill and others, 1992; Van den Elsen and others, 1995; Wang and others, 1996).

### *2.2.1 Conclusions*

Image data fusion refers to the ability to correlate information present in images from more than one sensor modality. Different approaches and algorithms are applied to satisfy specific problem constraints. For all approaches, the first step in fusing the information is to extract features from each of the sensor images. Once similar features are determined from each sensor image, the association is direct.

Past efforts in the area of data fusion have not provided a generic algorithmic approach for extracting features or for predicting one sensor image from another. The first step in analyzing two sensor images is feature extraction. The next section discusses the development of biological computational models that were inspired by the computational powers of the human nervous system.

## *2.3 Design Principles of Neural Systems*

Digital computing is in its infancy, compared to the power and capabilities of the human biological neural systems -- the brain, and the nervous system. The biological computer has an incredible capacity and is extremely effective in processing sensory inputs and, from those inputs, interacting with the environment (Tank and Hopfield, 1993). The biological computer's effectiveness inspires researchers to attain this computational power artificially, through programming and digital computers and by studying the principles and designs of the human neural systems.

Study of the human neural systems has been extensive. One of the primary sensors of the biological neural system is the visual system. Much study has been conducted in this area and models developed for mathematical simulation of the automatic processes performed by the human eye (Daly, 1993; Field, 1990; Kabrisky, 1966; Maher, 1970; Martin, 1996; McLachlan, 1962). To understand what is received from the sensory inputs, modeling from the most basic of brain units -- the neuron -- has been implemented in computers and these algorithms are called neural networks. Neural networks are electronic circuits that attempt to mimic the brain functions that process information. The following sections discuss some of the design principles of biological neural systems and the mathematical modeling of those designs. Specifically, the human visual system and the biological neuron will be described, and the models that appear to simulate the activities of these neural systems will be discussed.

### *2.3.1 Mathematical Modeling of the Human Visual System*

The human's powerful abilities in recognizing and interpreting shapes and forms and performing pattern recognition and classification provide the inspiration for mimicking these abilities computationally. The primary sensory input to the brain for imaging is the visual system, which has been studied extensively. Although attempts to understand human perception have been in progress for over a hundred years, the current concepts were developed in only the last few decades (Zeki, 1993 and 1993a). Before that, much original thought involved the concept that objects emit visual codes that are "impressed" on the retina, analogous to the photographic process. This was consistent with a dualistic philosophical doctrine that persisted, namely, that sensing and understanding were separate faculties, each utilizing a distinct seat within the cortex

(Zeki 1993 and 1993a). Today, however, the understanding of the visual system and perception is much more integrated, although complete understanding has not yet been achieved.

A brief summary of the structure of the human visual system follows. Figure 2 shows the visual-system pathway through the brain to the cortical area. Reflected light is received at the eye. The structure of the eye includes a cornea, aqueous humor, pupil, lens, vitreous humor, and retina. The optics of the eye attenuates high spatial frequencies in the input signal. At the back of the eye, the filtered optical signal is sensed by a multi-layer array of neurons in the retina. The retina has coarse- and fine-resolution sampling capability. A small area of the retina called the fovea is the high-resolution, fine-focus area of the retina. In the foveal region, a one-to-one correspondence exists between the received optical signal and the transmission of that signal through the optic nerve and back to the primary visual cortex. The off-foveal region is the coarse-resolution area of

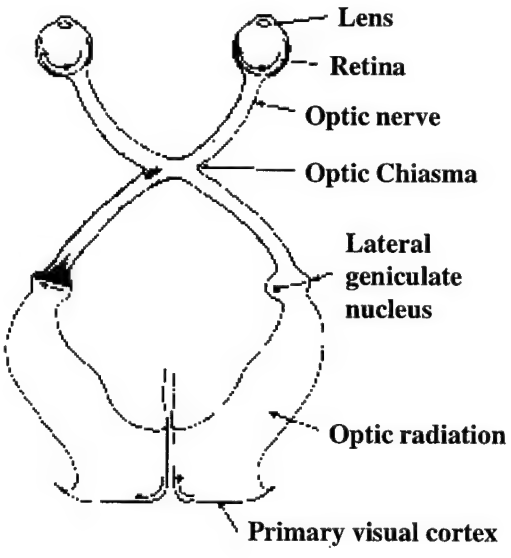



**Figure 2. Visual-System Pathway (Baron, 1987).**

the retina, and signals are multiplexed through a limited number of optic-nerve fibers and transmitted back to the primary visual cortex.

Efforts during this century to understand the functionality of the visual system were spearheaded by W. J. S. Krieg (1953), H. G. Dusser de Barenne and others (1942), and W. S. McCulloch (1951). Krieg developed a dissection and reconstruction technique for diagramming the major pathways in the visual system. About the same time, Dusser de Barenne and McCulloch developed the technique of neuronography for determining the direction of information flow in the pathways. This work validated the research of Krieg, and the combination of the two techniques was instrumental in providing understanding of the one-way information flow from the eyes to the primary visual cortex as well as the complex interconnections between the primary visual cortex and the cortical regions of the brain (Kabrisky, 1966).

As research continued, it was found that from the primary visual cortex, one input is connected to many points in the cortex and that many points in the primary visual cortex are connected to one point in the cortex. Figure 3 is a one-dimensional (1-D) representation of these one-way information pathways through the brain. The areas of interconnection are localized in both cases. This combination of points results in a single output. Understanding the method of interconnection of the cortical regions furnished the impetus for development of a transformation-model representation of the human visual system called the Kabrisky model (1966). This model postulates the connectivity from the primary visual cortex to a cortical area of the brain and back to a single point, enabling the system to perform a two-dimensional (2-D) cross correlation (Kabrisky, 1966).

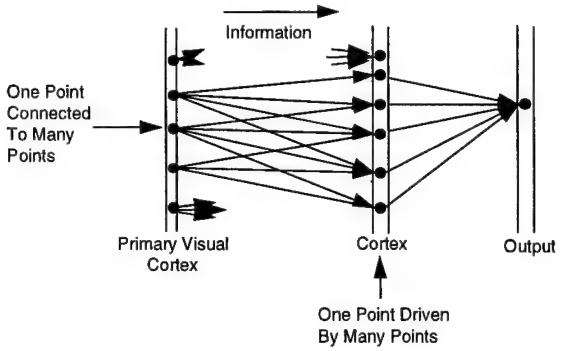



**Figure 3. 1-D Representation of Information Pathways through the Brain (Kabrisky, 1966).**

Mathematically, the Kabrisky model implies a transformation between the input at the eye and outputs at the various cortical stages. This model prompted studies showing that certain form estimates and form-detection behavior of the human visual system can be accurately modeled by assuming that the human visual system employs Fourier-like transformations of visual data (Kabrisky, 1966; Maher, 1970; McLachlan, 1962). Additionally, experiments have quantified a high correlation between subjective evaluations of the similarity between images and those images in Fourier space (Maher, 1970). Section 2.3.1.1 discusses concepts of the human visual system that have been modeled by Fourier analysis.

Other properties of the human visual system have been identified. As described above, the optical-information signal received at the eye is transformed and transmitted to the visual cortex for interpretation. The visual system also generates additional transformations by processing the input signal based on location within the field of view. Data received from the left side of the field of view in each eye are projected to the right-hemisphere primary visual cortex, and those received from the right side are projected to the left-hemisphere primary visual cortex (Zeki, 1993). In addition, a logarithmic spatial

transformation occurs that incorporates scaling of distances and radial locations (Baron, 1987).

The objective of the current research has been to associate features that represent the same object and, therefore, appear to the human observer to be similar but digitally are represented as being significantly different. The high correlation of similar Fourier images provides motivation to examine the use of Fourier space for feature extraction. However, Fourier analysis does not provide a complete answer since the frequency information in one location of an image is spread throughout the entire Fourier output and a local characteristic of the image becomes a global characteristic of the transformed image. Also the Fourier transform is sensitive to shifts and rotation in the image being analyzed.

Examination of Fourier analysis has led to investigations with other transforms -- specifically, wavelet transforms (Field, 1990; Hubbard 1996). Wavelets are designed based on the varying-scale issue in which the prototype wavelet function, called the mother wavelet, is applied on different scales. Some wavelets are sensitive to rotation; unlike in Fourier analysis, however, wavelets provide orientation information that can be used to determine and compensate for rotation. Whereas Fourier analysis transforms an image from a function of one single variable (e.g., time or space) to a function of another single variable (e.g., frequency), wavelets transform a function of one single variable to a function of two single variables (e.g., time and scale or space and scale). Wavelets are perceived as an effective means of mimicking the human visual system because of the multiresolution, multispatial concept of going from lower resolution for the coarse overall shapes to increasingly higher resolution for finer detail (Field, 1990). Section 2.3.1.2

contains a discussion of wavelet analysis as it applies to modeling concepts of the human visual system.

### *2.3.1.1 Fourier Analysis*

The Kabrisky model -- that certain form estimates and form-detection behavior of the human visual system can be modeled by Fourier-like transformations of visual data -- provides motivation to examine the use of Fourier space for extracting features. Mathematically, the 1-D Fourier transform of a function *f(x)* is defined as:

$$F(u) = \int_{-\infty}^{+\infty} f(x) e^{-j2\pi ux} dx \tag{1}$$

The inverse Fourier transform of *F(u)* is defined as:

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(u) e^{j2\pi ux} du \tag{2}$$

These two equations are called the Fourier-transform pair and can be shown to exist if *f(x)* is absolutely integrable and *F(u)* is absolutely integrable. These conditions are almost always satisfied in practice (Bracewell, 1965).

Black and white image data is a matrix of numerical values (pixel values) that represent the relative gray scale of different entries in the matrix. Processing image data involves only discrete, uniformly sampled values. For these cases the discrete Fourier transform (DFT) is appropriate. The 2-D DFT pair is given by:

$$F(u_m, v_n) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{i=0}^{N-1} f(x_i, y_k) e^{-j2\pi (x_i \cdot u_m + y_k \cdot v_n)/N^2} \tag{3}$$

for $m = 0, 1, \ldots, N-1$, and $n = 0, 1, \ldots, N-1$

$$f(x_i, y_k) = \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} F(u_m, v_n) e^{j2\pi (u_m \cdot x_i + v_n \cdot y_k)/N^2} \tag{4}$$

for $i = 0, 1, \ldots, N-1$, and $k = 0, 1, \ldots, N-1$

Maher (1970) has shown that using Fourier analysis to model the human visual system effectively correlates to a subjective human evaluation and comparison of forms. The experiment involved 40 subjects who quantified their perception of the similarity of images of animal crackers. The subjects were able to provide numerical values on a scale of 1 to 10 in reporting perceived similarities among 10 different images. A computer, using low-pass-filtered two-dimensional Fourier transforms, calculated the distances between the animal-cracker image patterns. The extent of correlation between the test-subject mean values and the Fourier outputs was calculated to be between 0.721 and 0.970, indicating a high correlation between the human outputs and the computer simulations. This work demonstrated a statistically significant correlation between human pattern perception and machine pattern classification using Fourier transforms (Maher, 1970).

The experimental work of Kabrisky (1966) and Maher (1970) showed that Fourier transformation of image data correlates with human perceptions and therefore the Euclidean distance between the Fourier transformation of two images is a measure of similarity between the images. The specific algorithm for testing for similarity between two images is as follows:

1. Generate the Fourier transform of each image.
2. Energy-normalize each pixel coefficient pair in the transformed image matrix by dividing that pair by the square root of the sum of the squares of all the elements in the matrix

$$\hat{F}(u_m, v_n) = Enorm\{F(u_m, v_n)\} = \frac{F(u_m, v_n)}{\sqrt{\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} F(u_m, v_n)}} \tag{5}$$

where $(u_m, v_n)$ represents the coefficient pairs associated with a pixel location.

3. Calculate the Euclidean distance between Image *f* and Image *g* using

$$Dist(F,G) = |F(u_m, v_n) - G(u_m, v_n)|^2 \quad (6)$$

where *F* is the Fourier transform of image *f*, *G* is the Fourier transform of image *g*, and the $(u_m, v_n)$ values correspond to transform coefficients for an image pixel location.

Since each image was normalized, the maximum distance between any two images will be two. Therefore, thresholds can be applied that determine similar and dissimilar sets.

Each image is now represented losslessly using all of the *(m, n)* coefficients. However, if a somewhat lossy representation of the original image can be tolerated, then only a subset of the coefficients requires analysis. To evaluate the lossy versus lossless criteria, an explanation of the Fourier transform values is necessary. The low-frequency Fourier coefficients near the DC term, corresponding to the low frequencies, contain the maximum information concerning the overall shapes and forms in the data. The higher frequency coefficients contain the very high frequency information that is found in edges and noise. In the case of images, many of the high-frequency Fourier coefficients can be eliminated, and the image will reconstruct with visually insignificant loss of detail (Bracewell, 1965; Pao, 1989). The distance formula given in Equation (6) can be employed effectively to discriminate different shapes or forms in the data when all of the Fourier coefficients are used or when only the dominant low-frequency Fourier coefficients of the original images are used. Close proximity of the coefficients, e.g., clustering, is a means to assess similar form or shape.

The four gray-scale images in Figure 4 demonstrate this concept of the Fourier-coefficient vectors clustering to similar shapes or forms and show that the distance metric provides an effective evaluation of similarity. The top two crosses are multi-gray-scale images; each cross has a different gray scale, and the gray scales are not linearly related. The bottom two triangular images have different gray scales that are not linearly related. These four images demonstrate that the Fourier coefficients recognize form or shape, even as the gray scale varies.

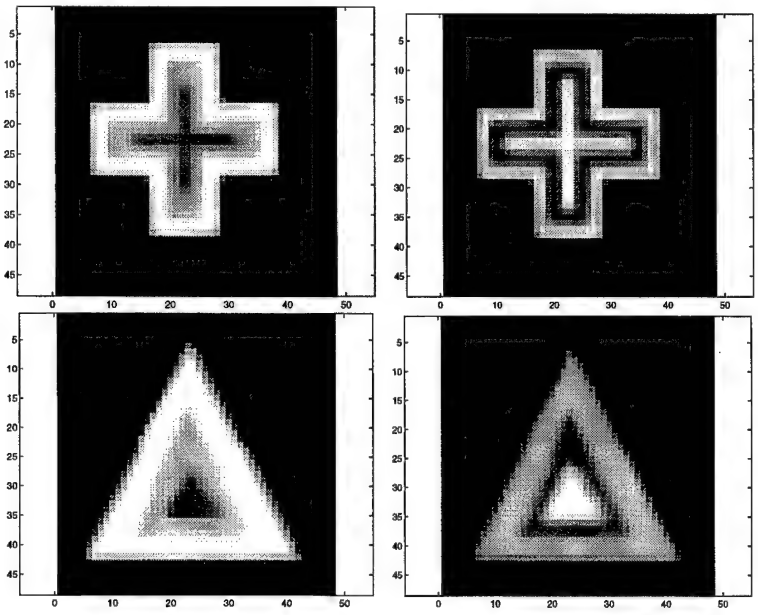


**Figure 4. Gray-Scale Images: Crosses, top; Triangles, bottom.**

Each of these images was transformed to Fourier space and the energy normalized according to Equation (5). The Euclidean distance between them was then calculated according to Equation (6). The data in Table 1 show the distances between the different images. As expected, the distance between two identical images is 0.0.

The top two images in Figure 4 are similar in pattern but vary with respect to gray scale. The image on the top left is a three-gray-scale cross and that on the top right is a cross having a different three-gray-scale. For these two images, the distance between the

| | Cross | "Inverse" Cross | Triangle | "Inverse" Triangle |
|---|---|---|---|---|
| Cross | 0.0 | 0.58398 | 1.38249 | 1.28647 |
| "Inverse" Cross | | 0.0 | 1.29478 | 1.19813 |
| Triangle | | | 0.0 | 0.485569 |
| "Inverse" Triangle | | | | 0.0 |

**Table 1. Distances Between Gray-Scale Images in Figure 4.**

two Fourier-transformed images is 0.58389. The two images on the bottom are triangles with different gray levels, and the distance between these images is less than 0.5. The distance calculated for the dissimilar shapes, i.e., cross to triangle is greater than 1.0.

Since the distance between the normalized Fourier-transformed images can have a maximum value of only 2.0, the distance values nearest 0.0 would indicate identical or very similar images and distance values near 2.0 would indicate completely dissimilar images. The data in Table 1 demonstrate this concept. Identical images have a distance of 0.0, and images of the same shape but different gray scale have a distance of much less than 1.0 -- in other words, values much nearer 0.0 than 2.0. Dissimilar shapes, i.e., crosses and triangles, have distances greater than 1.0 -- in other words, values much nearer 2.0 than 0.0. As shown experimentally (Maher, 1970), these distance values correlate with the way in which the human would view and quantitatively measure similarity in the images.

In summary, energy present in the Fourier coefficients clusters to similar forms and patterns: closeness in Fourier energy space corresponds to closeness or similarity in the human visual system. This is the case when all the Fourier coefficients in each image are used or when those dominant low-frequency Fourier coefficients are used that can

effectively distinguish the different shapes or forms in the data. Fourier-transformed images can be used to extract similar features in the same way the human judges features to be similar.

#### *2.3.1.2 Wavelet Analysis*

Evidence has been presented that the human visual system performs a wavelet-like transformation to represent the visual environment (Field, 1990). Wavelets are a mathematically based intermediate compromise of the capability of the human eye to perform Fourier analysis of the visual scene as well as respond to and select specific edges (Field, 1990). Whereas the 1-D Fourier transform maps a signal from a function of one single variable (e.g., time or space) to a function of another single variable (e.g., frequency), a wavelet transform maps a function of one single variable to a function of two variables (e.g., time and scale or space and scale). The discrete wavelet transform can be viewed as filters and perform the same signal-analysis function as Pyramid Algorithms in the case of image processing (Burt and Kolczynski, 1993; Hubbard, 1996; Toet and others, 1989) and Subband Coding in the case of signal processing (Hubbard, 1996; Mallat 1989; Mallat and Zhang, 1993; Smith and Barnwell, 1986). The underlying commonality is the application of a succession of filters (Hubbard, 1996). As a result wavelets cannot be physically correlated in the manner of Fourier and, therefore, are more difficult to interpret in a physical sense (Field, 1990).

By application of a succession of filters in the multiresolution theory, wavelets have been linked to the filters used in signal processing. Multiresolution refers to breaking a signal into a coarse-resolution image that provides the lowest frequency components of the signal and then into progressively higher resolution images that

provide the finer details of the signal. For standard wavelets, based on a dyadic system, each progressively "higher" resolution is referred to as an octave; thus, from one resolution to the next, the resolution is twice as fine and doubles the frequency of the wavelets to allow them to encode frequencies that are twice as high. Wavelets are of constant shape and are changed by dilating and translating one function. This transformation results in resolution, scale, and frequency changes occurring simultaneously (Hubbard, 1996).

The 1-D wavelet transform is defined starting with the "mother" or analyzing wavelet. The mother wavelet, $\Phi(x)$, is dilated (or scaled) and translated to define an orthonormal basis. The orthonormal basis, $\{\Phi_{(j,l)}(x) | \, j \in \Im, l \in \Im\}$, is formed by $\Phi_{j,l}(x) = 2^{\frac{-j}{2}} \Phi(2^{-j} x - l)$, where the indices $j$ and $l$ are integers that dilate and translate the mother wavelet, respectively. The scale index, $j$, indicates the wavelet width, and $l$ is the location index that indicates the position which controls the translation or shift of the wavelet. The mother wavelet is defined to have an area of zero; therefore, as the scale index $j$ increases, $\Phi_{j,l}(x)$ becomes wider and shorter, and the translation step increases. as the scale index $j$ decreases, $\Phi_{j,l}(x)$ becomes thinner and taller, and the translation step decreases.

To span the data in the domain, a scaling equation, based on the mother wavelet, is applied at different resolutions

$$\phi(x) = \sum_{k=-1}^{N-2} (-1)^k c_{k+1} \Phi(2x + k) \tag{7}$$

where $\phi(x)$ is the scaling function for the mother wavelet $\Phi(x)$, and the $c_k$'s are the wavelet coefficients (Graps, 1995). The wavelet coefficients, $\{c_k\}$, are chosen to satisfy both linear and quadratic constraints of the form $\sum_{k=0}^{N-1} c_k = 2$, $\sum_{k=0}^{N-1} c_k c_{k+2l} = 2\delta_{l,0}$.

Considering the set of wavelet coefficients, $\{c_0, \cdots, c_n\}$, as a filter vector, a transformation matrix can be generated and applied to raw data. Two dominant filter patterns or two sets of wavelet coefficients -- one that functions as a smoothing filter and one that extracts the detailed information in the data -- make up the transformation matrix (Graps, 1995). The smoothing or low-pass filter *H=h(n)* is defined as

$h(n) = \frac{1}{2}\left\langle \phi\left(\frac{x}{2}\right), \phi(x-n)\right\rangle$, where $\phi(x)$ is the scaling function defined in Equation (7).

The detail filter or high-pass filter, *G=g(n)*, is defined as $g(n) = (-1)^{1-n} h(1-n)$.

A computationally efficient method of implementing the wavelet transform is the hierarchical pyramid algorithm. This algorithm operates on a finite set of input data of length N, where N is a power of two. A transformation matrix is generated by alternating rows of the smooth and detail filter coefficients, *h(n)* and *g(n)*, respectively. The matrix is first applied to the original full-length data vector, creating a "detail" data vector. The original data vector is smoothed and decimated by half, and the matrix is applied again, creating another detail data vector. This process continues on the smoothed, decimated data until a trivial amount of data remains in the smoothed vector. The output of the transformation consists of the accumulated detail vectors and the remaining smooth vector.

Figure 5 graphically depicts application of the wavelet transform using the pyramid algorithm on 2-D image data. The *H* and *G* filters are applied to the image in both the horizontal and the vertical direction, and the resultant filter outputs are subsampled by a factor of two. This generates three orientation-selective high-pass subbands, $f_{HighHigh}$, $f_{HighLow}$, and $f_{LowHigh}$, and the low-pass subband, $f_{LowLow}$, as shown in Figure 5. Since most of the information is present in the $f_{LowLow}$ subband decomposition, this subband is utilized for further decomposition. Repeating the process on the $f_{LowLow}$ subband generates the next level of the decomposition; this continues until the desired level is reached. Each decomposition creates four filter subbands and is referred to as an octave.

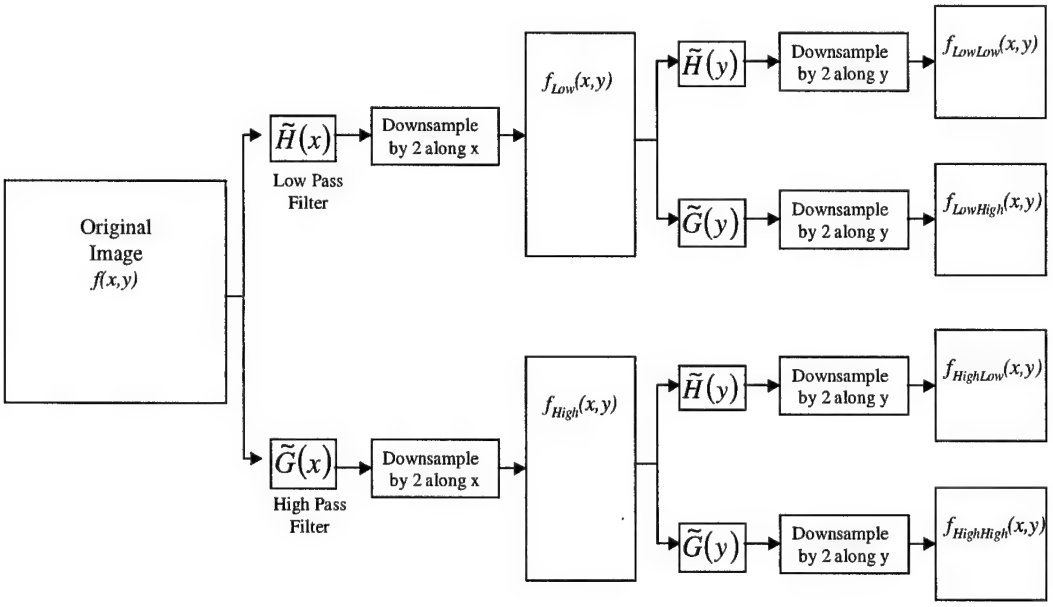



**Figure 5. Wavelet Transform.**

In choosing a particular wavelet, the following criteria must be considered.

- Compact Support. Wavelets with compact support are not infinite and have the value of zero outside a certain interval.

- Number of Vanishing Moments. The moment $k$ of a function $f$ is the integral of that function multiplied by its variable raised to the power $k$

$$m_k = \int_{-\infty}^{\infty} f(x)x^k dx \tag{8}$$

The $k^{th}$ moment vanishes when this integral is zero; since the wavelet and the function are orthogonal, their scalar product is zero. The number of vanishing moments determines what the wavelet does ***not*** see. For example, one vanishing moment refers to a wavelet that is "blind" to linear functions, and two vanishing moments refers to a wavelet blind to quadratic functions, etc. (Hubbard, 1996). The number of vanishing moments determines the convergence rate of wavelet approximations of smooth functions. If the image is smooth, an increase in the number of vanishing moments leads to smaller wavelet coefficients. On the other hand, if the image is non-smooth, an increase in the number of vanishing moments leads to larger wavelet coefficients. However, the number of vanishing moments needed and whether the moments must vanish entirely or only essentially vanish depend on the application (Hubbard, 1996).

- Length of Filters. The length of the filter is defined by the number of coefficients present for application to the data. Obviously, short filters are preferable, but a trade-off exists between short filter length and the smoothness and number of vanishing moments.

For image-compression applications, the low-pass-filter output generally contains most of the information content of the image. Another important consideration, specifically for wavelet-based compression, is finding the mother wavelet that causes the high-pass-filter terms to approach zero. With wavelet selection such that the high-pass-

filter terms approach zero, the high-pass-filter outputs can be discarded without significantly affecting the visual quality of the image reconstructed from the low-pass-filtered output.

In summary, wavelets provide low-frequency information which is similar to that obtained with Fourier analysis. In addition, wavelets can provide segregated high-frequency information, often called edge images, based on the specific wavelet that is used. Wavelets can be envisioned as a combination of low-frequency shape or form detectors, such as provided by Fourier analysis, along with a high-frequency edge-detection capability. Wavelets also satisfy some of the human-visual-system concepts that are well understood and have been modeled.

#### *2.3.1.3 Visual-Difference Predictor*

One application of human-visual-system concepts is determining a quantitative value for image quality. The two types of image quality are absolute, which refers to assessment of an image based on the image itself, and relative, which refers to assessment of an image relative to a reference image. The term fidelity or faithfulness to the original, refers to relative quality between two images.

Extensive research has been performed in the area of measuring perceptual-image fidelity using a human-visual-system modeling approach (Daly, 1993; Heeger and Teo, 1995; Martin, 1996; Westen and others 1995). These efforts were based on approaches that produce a map specifying the perceptual differences between the two images; this provides the ability to determine specific locations in the image that are similar and dissimilar. Recently, several approaches have been used in quantifying the perceptual

image fidelity of two images. One such approach is the Visual-Difference Predictor (VDP) (Daly, 1993; Martin, 1996).

One of the key tools of the VDP is the Gabor transform (Daly, 1993). Gabor expansions are used in image processing because they reveal the time-frequency distribution of the image (Pei and Yeh, 1997). A Gabor transform is basically a windowed Fourier transform, with the envelope for the window being a Gaussian. For this analysis, the size of the window is fixed, but the frequencies inside the window vary. For this transform the variables are the frequency and the position of the window. The smaller the window, the more precise the spatial information obtained; however, some low-frequency information is lost. The larger the window, the more frequency information obtained; however, the information concerning time is less precise (Hubbard, 1996). The Gabor transform is basically a series of band-pass filters that cover the entire image, and the size of the window determines the predominant precision in the frequency or time domain.

A basic processing structure for fidelity measurements of two images is depicted in Figure 6 (Martin, 1996). The images are processed by a psychophysical human-visual-system model in which the retinal stage processes the image by an amplitude nonlinearity and then a contrast-sensitivity filter (CSF), which is a linear filter. The cortical stage follows the retinal stage and consists of a spatial-frequency transform across multiple frequency bands called the cortex bands. These bands are calculated using Gabor filters as the means of achieving band-pass filtering. This is followed by differencing, masking, and detection operations within each cortex band, which constitutes the comparative

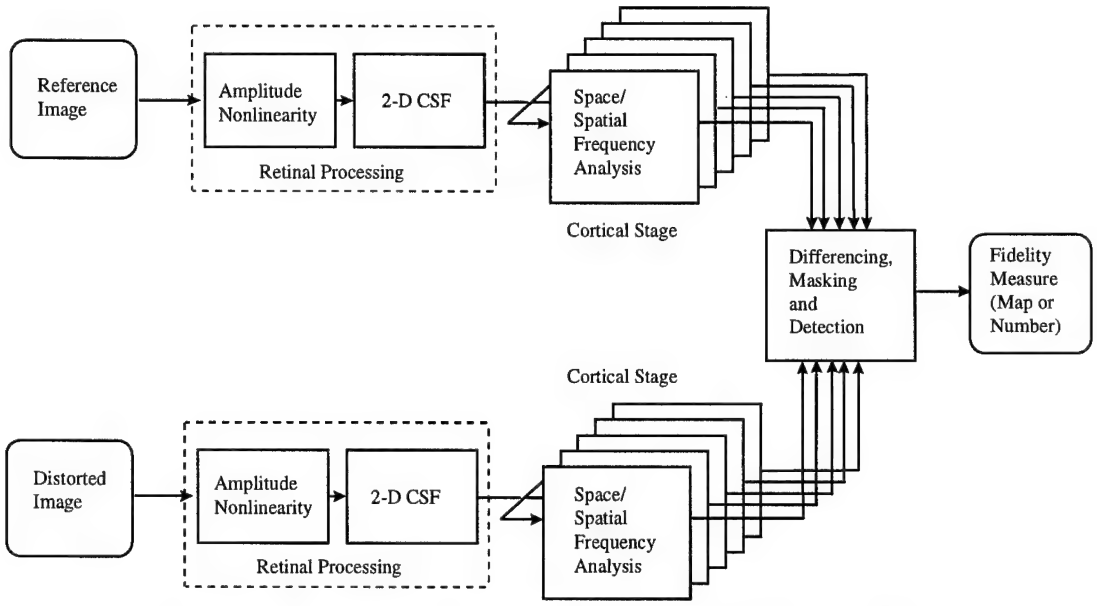



**Figure 6. Processing Structure for the VDP (Martin, 1996).**

stage of the two images. The resulting output -- the VDP algorithm -- is a map or matrix of fidelity measurements of the two images being compared.

The VDP algorithm predicts the probability of detecting a pixel of contrast *c*. The point at which contrast *c* is detectable is the threshold where a human would subjectively decide that the pixels of one image are not the same as those of another. This VDP fidelity measure is a pixel-by-pixel map, with each map entry being a quantitative metric for evaluating the similarity of two images at each specific pixel location. This map highlights locations where the associated pixels of the images are similar as determined by a human perceptual threshold. This algorithm has extended the ability to measure relative image quality or image fidelity from the global level to the pixel level.

#### *2.3.1.4 Conclusions*

The concepts of the human visual system provide a foundation for reducing image data to the pertinent features of concern and a means of quantifying image quality. Fourier and wavelet transforms can be applied to extract the features of interest and are

based on an understanding of the human visual system. Once the image data has been processed, the VDP can be utilized to assess the quality of an image relative to a reference image.

### *2.3.2 Neuronal Modeling with Artificial Neural Networks*

The human brain possesses remarkable processing power. It interprets imprecise information from the senses at an incredible rate. Moreover, it learns, without any explicit instructions, to create the internal representations that make interpretation possible (Hinton, 1993).

Scientists have attempted to understand and mimic the vast capabilities of the human brain. One attempt to simulate the brain's learning processes is to model the work of individual neurons. An understanding of the function of the biological neuron is the foundation for the concept of neural networks. In the human brain, a typical neuron in an excited state conveys information to other neurons by creating action potential (~ 70 mV) which travels down branches called axons. The receiving neuron collects signals through branches known as dendrites. At the connection, a structure called a synapse converts the action potential from the axon into chemical ions that excite or inhibit activity in the connected neuron. Learning occurs by changing the effect of the synapses in such a way that their influence among neurons is altered (Fischbach, 1993).

The first gross-idealization model of an individual neuron was the single perceptron (Rosenblatt, 1959). Each perceptron has inputs that are weighted to simulate the biological synaptic weights. The weighted inputs are then summed, a bias term is added, and a transformation, called an activation function, is applied to the weighted sum. A diagram of a single perceptron is shown in Figure 7. A single perceptron can be used

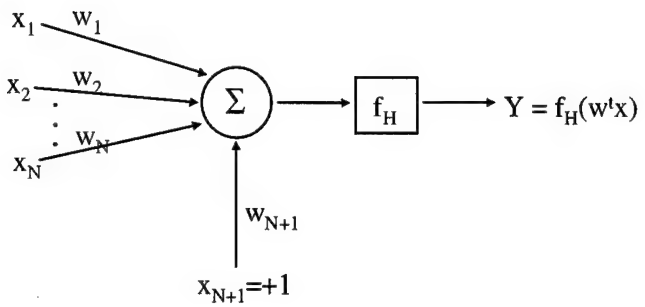



**Figure 7. Diagram of Perceptron.**

to discriminate linearly separable classes or make the distinction between two pattern classes having linear separation.

A multi-layer perceptron can be generated by adding an intermediate layer between the input and output nodes. For not linearly separable (but hyperplane or multiple hyperplane separable) data, multi-layer perceptrons are required for discriminating various classes. The single-perceptron model is used as the foundation for generating a multi-layer perceptron; the architecture of the latter is shown in Figure 8. The weights are designated by subscripts to specify the connection between the

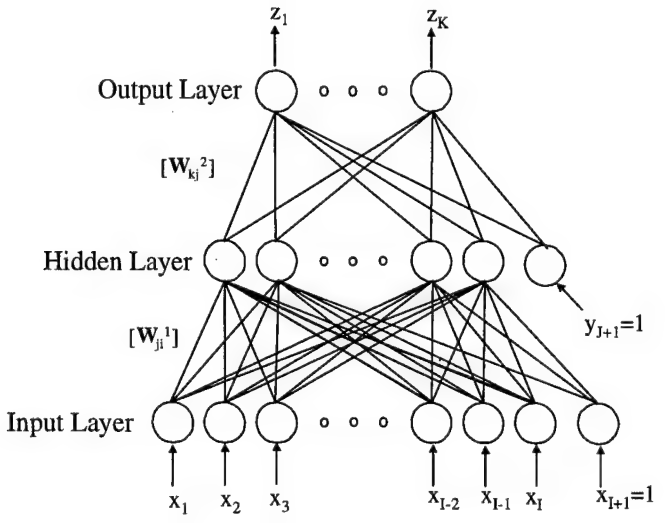



**Figure 8. Multi-layer Perceptron Artificial Neural Network.**

individual input, hidden-layer and output nodes, and the superscript designates the weight layer, e.g., 1 for first layer, 2 for second layer. Each output from the output layer is designated by $z$.

In the computational implementation, a multi-layer perceptron consists of interconnected units called nodes; each node serves as a model for an individual neuron, with weights being applied to each interconnection. As in the single-perceptron case, each node receives a series of inputs, and each input is weighted. For each node, the weighted inputs are summed; a bias term is added; and a transformation is applied to the weighted sum. From this simple, gross-idealization neuron model, interconnections among the neuron nodes are generated and weighted to create a large interconnected network that is capable of learning and classifying nonlinearly separable data. A multi-layer perceptron is a feedforward (data flows in one direction only) artificial neural network (Hinton, 1993).

Artificial neural networks are universal approximators and, therefore, useful for solving various problems related to distinguishing classes or patterns (Cybenko, 1989). These networks provide a representation of a general continuous function of an n-dimensional real variable, $x \in \Re^n$, through a finite linear combination of the form:

$$\sum_{j=1}^{N} \alpha_j \cdot \sigma\left(y_j^T \cdot x + \theta_j\right) \tag{9}$$

where $y_j \in \Re^n$ and $\alpha_j, \theta \in \Re$ are fixed ($y^T$ is the transpose of $y$; thus, $y^T x$ is the inner-product of $y$ and $x$) (Cybenko, 1989). The activation function, $\sigma$, of each of the nodes of the neural network is of primary concern since it must meet certain conditions (although mild) to provide universal approximation power. The primary condition is that $\sigma$ must be

bounded at $-\infty$ and $+\infty$. Under this mild condition, and $\sigma$ is continuous, the linear combination sums in Equation (9) will be dense in the space of continuous functions on the unit cube and, thus, can provide the universal approximating power.

#### *2.3.2.1 Autoassociative Neural Network (AANN)*

A special architecture of a feedforward multi-layer perceptron neural network is the Autoassociative Neural Network (AANN). Initially, AANNs were used for image compression and, in subsequent research, for image processing and feature extraction (Cottrell and others, 1989; DeMers and Cottrell, 1993; Hecht-Nielsen, 1990 and 1995; Kramer, 1991; Turk and Pentland, 1991).

AANNs are a type of self-organizing map of input images. Such a map is used to train the neural network in an unsupervised manner. In an AANN, the input and the target outputs are the same images, with a reduced dimension on the hidden layer which provides data compression. Originally, the neural networks evaluated were those with linear activation functions, and these networks provided the principal components of the image being compressed (Baldi and Hornik, 1989; Oja, 1992). Subsequent research employed a multilayer, nonlinear neural network that compressed and reconstructed the original data; this network was based not on principal components of a linear subspace but on projections to a nonlinear submanifold of the feature space (Cottrell and others, 1989; DeMers and Cottrell, 1993; Hecht-Nielsen, 1990 and 1995; Kramer, 1991; Malthouse, 1996). Applications of AANNs include data compression, with the compressed representation being used as features for further analysis. The compressed data can be used in many ways, e.g., for classification, as a lossy storage medium, and as a means of transmitting a data set over transmission lines (Cottrell and others, 1989;

DeMers and Cottrell, 1993; Hecht-Nielsen, 1990 and 1995; Padgett and others, 1998; Turk and Pentland, 1991).

For feature extraction or data compression, a typical AANN has at least three hidden layers. In a three-hidden-layer AANN, as shown in Figure 9, when the middle hidden layer of the network, called the bottleneck, has fewer nodes than the input and output layers, then data compression is achieved in a self-organizing fashion.

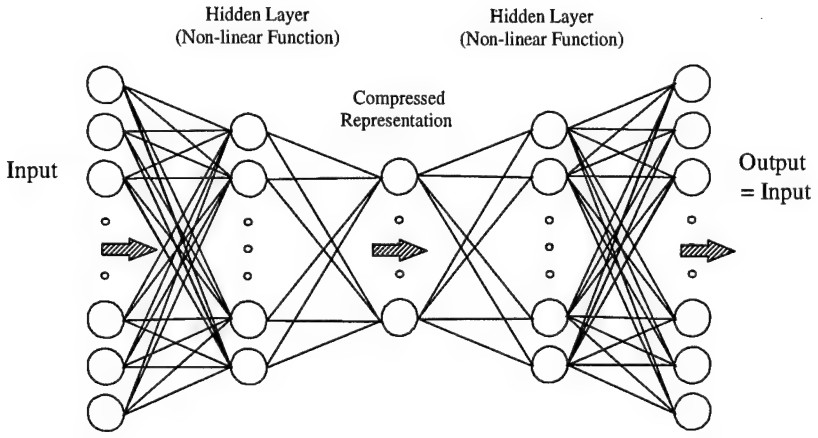



**Figure 9. Typical Autoassociative Neural Network.**

Data compression is achieved in the AANN by processing a data set with an ambient dimension of $n$ to a reduced manifold with an intrinsic dimension of $m$, where $n>m$. Here the AANN is used to reduce the object residing in an $n$-dimensional space to its intrinsic dimensionality by mapping the input data to the reduced space and then back to the original data. A specific implementation of the data-mapping concept of the AANN is equivalent to vector quantization (Hecht-Nielsen, 1995). The vector-quantization implementation, is discussed in Section 2.3.2.1.2, uses a special activation function on the middle hidden layer to provide the "codewords" of the input data and the partition sets associated with each codeword. This function was developed to provide

higher compression rates and to generate a means of comparing equivalence relationships in AANNs and other statistical clustering methods. The following two subsections describe data-mapping and vector-quantization concepts of AANNs.

*2.3.2.1.1 Data Mapping*

In data mapping, the AANN generates the functional mapping of the input data vectors to a reduced representation and then back to the target data vectors. The concept of data mapping has been referred to as the encoder/decoder problem (Kramer, 1991). For the typical AANN shown in Figure 9, the input and the first and second (middle) hidden layers are considered to be the encoding portion of the network. The second (middle) and the third hidden layers and the output are considered to be the decoding portion of the network (Kramer, 1991).

For encoding, the data are mapped from the input data vectors having an ambient dimensionality to a compressed/reduced representation having intrinsic dimensionality on the middle hidden layer. Notationally, the encoding portion of the AANN maps the input data vector $x$ having dimension $n$ ($x \subset \Re^n$) to the reduced dimensional space, $\alpha$; this generates a mapping function $f$ which approximates the mapping from the original data to its intrinsic dimensionality $m$ ($f : \Re^n \rightarrow \Re^m$) where the compressed representation is $\alpha \subset \Re^m$. Here, the assumption is made that $n > m$ which ensures that the middle hidden layer is a compressed representation.

In instances in which the reduced representation of the data is the ultimate goal of using the AANN (Cottrell and others, 1989; DeMers and Cottrell, 1993; Hecht-Nielsen, 1990 and 1995; Padgett and others, 1998; Turk and Pentland, 1991), the middle hidden layer is an "output" for the application at hand. This encoding portion of the AANN can

be viewed as a single-hidden-layer, feedforward neural network. With a nonlinear activation function on the first hidden layer, such as a sigmoid or a hyperbolic tangent (tanh) provides universal approximation power (Cybenko, 1989 and 1996).

The second and third hidden layers and the output layers are considered to be the decoding portion of the AANN. These layers generate a mapping function, $g$, which approximates the mapping from the reduced representation of the data of dimension $m$, to the target data (identical to input-data vectors) ( $g : \Re^m \rightarrow \Re^n$ ). Thus, the AANN provides multiple data mapping, $x \xrightarrow{f} \alpha \xrightarrow{g} x$, as shown graphically in Figure 10.

It has been shown that the typical AANN provides a compressed representation on the middle hidden layer. The various imaging applications of this network require compressed representations, i.e., $n > m$, for computational efficiency or class separation (Padgett and others, 1998; Turk and Pentland, 1991). Use of AANNs for data compression implies an underlying assumption—namely, that the intrinsic dimensionality of the data ***is*** less than the ambient dimensionality of the input data vectors, $n > m$. Only under this assumption will the output of the AANN be a lossless reconstruction of the original data. This means that the mappings, $f$ and $g$, are guaranteed to be lossless when the hidden-layer dimension, $m$, is the underlying intrinsic dimensionality of the original data (Kirby and Miranda 1994 and 1998). For dimension reduction, the assumption is made that $m < n$; however, if this is a required condition, the AANN may not be entirely lossless. The value of $m$ is based on the intrinsic dimensionality of the data and is not guaranteed to be less than $n$.

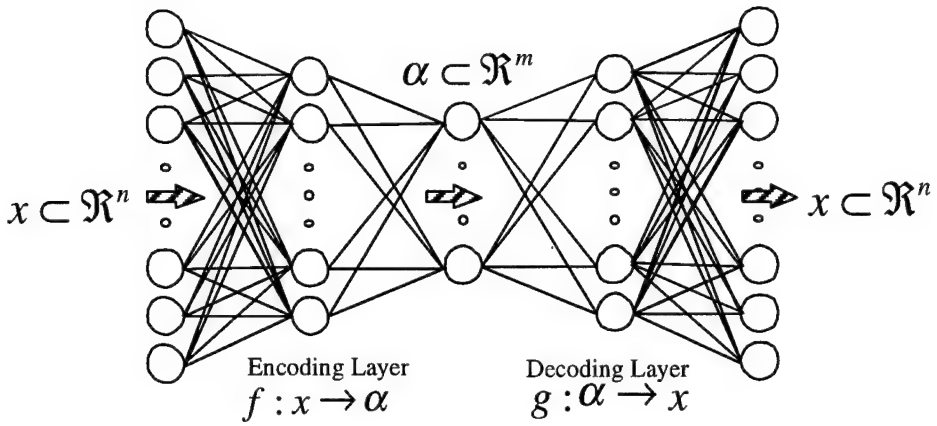



**Figure 10. Multiple Data Mappings of the AANN.**

Reviewing the concepts of functional mappings and the intrinsic dimensionality of the data, the conclusion is drawn that the AANN generates encoding and decoding basis sets which are spanned by the projections on the middle hidden layer. The basis subspaces are within the network architecture and are generated by the training of the AANN. Section 2.3.2.1.3 contains a discussion of the mean-squared error (MSE) as the AANN error function which generates a basis subspace in the encoding portion of the network that minimizes to a truncated eigenspace (Malthouse, 1996). This is an extrapolation of efforts with single-hidden-layer networks having linear activation functions in which it was shown that the network does provide principal components analysis (PCA) (Oja, 1992). However, the use of nonlinear activation functions in the hidden layers of the AANN provides a compressed representation on the middle hidden layer nonlinearly, which is a projection of the input data onto the truncated eigenspace. For this reason AANNs are often referred to as "nonlinear" PCA, because of the use of a nonlinear activation function (Cottrell and others, 1989; DeMers and Cottrell, 1993; Hecht-Nielsen, 1990; Kramer, 1991; Malthouse, 1996).

*2.3.2.1.2 Vector Quantization*

The AANN compresses the initial input to a reduced representation on the middle hidden layer. However, the numerical values of the nodes on this layer are continuous, and the number of bits required to encode these values has not been reduced. Vector quantization is a means of associating data samples with specific codewords; the data are further compressed using the index to the codeword (Linde and others, 1980).

The work of Hecht-Nielsen (1995) has demonstrated that a specific AANN architecture can be viewed as real-time coders -- devices that map each input data vector to a code. The specific AANN utilizes a three-hidden-layer network with tanh activation functions on the second and third hidden layers, a linear activation function on the output, and a special activation function on the middle hidden layer. This architecture is equivalent to vector quantization (Hecht-Nielsen, 1995).

What is required is an AANN with $m$ middle hidden layer nodes, where each of these nodes take on one of $N$ discrete output values. This will generate an AANN with $N^m$ number of codes on the middle hidden layer. The activation function that performs the vector quantization in the AANN is of the form

$$[y2]=\frac{1}{2}+\frac{1}{2(N-1)}\cdot\sum_{j=1}^{N-1}\tanh\left[a\cdot\left([w^2]\cdot[y1]-\frac{j}{N}\right)\right] \tag{10}$$

where $w^2$ is the second layer weight matrix, and $y1$ is the vector output from the first hidden-layer nodes. This function maps the input to each middle-hidden-layer node to one of the "stairstep" values. The value $a$ determines the steepness of the each stairstep (Hecht-Nielsen, 1995). Figure 11 shows the middle-hidden-layer activation function for $N = 10$ and $a = 200$.

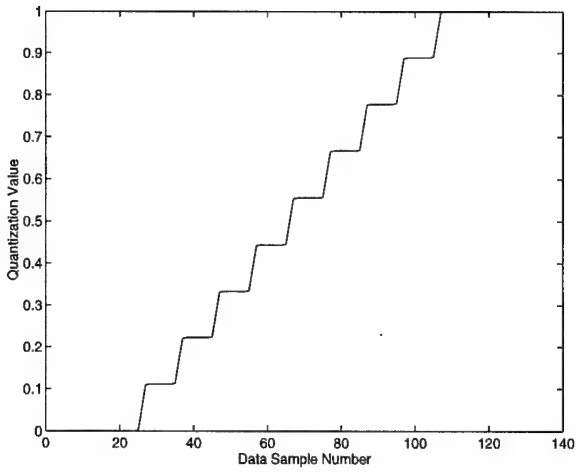



**Figure 11. Special Tanh Activation Function for $N$=10 and $a$=200.**

This smooth stairstep activation function essentially quantizes the middle-hidden-layer output to a few specific values that are equivalent to indices of codewords. (Codewords can be found by an iterative method that terminates in a local minimum when the average distortion, based on the distance from the codewords to the data points within their partition, no longer changes significantly. This can be a time consuming effort. Using AANNs provides a potentially practical means for efficiently constructing the codewords from complicated input data.) As $N$ approaches infinity, this stairstep approaches the ramp function (linearity) between 0 and 1. This network provides additional assistance in the compression process by quantizing data on the bottleneck. The number of quantization steps is selected by the problem being solved, which is based primarily on the amount of loss that is tolerable in the reconstructed image. Obtaining a lossless reconstruction means that an infinite number of codewords must be used, as $N$ approaches infinity, and therefore, the activation function on the bottleneck is reduced to a ramp function. Because the continuous values on the hidden-layer nodes have been discretized, this quantization technique becomes equivalent to vector quantization. Each of the discrete stairstep vectors on the middle hidden layer has a partition set of input

vectors associated with it. Thus, the discrete vectors on the middle hidden layer can be viewed as codewords to a partition set of data associated with them. This special activation function on the compressed middle hidden layer was developed to provide higher compression rates as well as to generate a means of performing equivalence relationships between AANNs and other statistical clustering methods.

*2.3.2.1.3 Error Function*

The training of the AANN involves development of a map between the input and the output layers. One systematic method for training multilayer artificial neural networks is backpropagation (Pao, 1989; Wasserman, 1989). The backpropagation algorithm is a deterministic training method that employs a type of gradient descent; that is, it follows the slope of the error surface downward, constantly adjusting the weights toward a minimum. The error surface can be highly varying, with many hills and valleys, and a sequence of weights can become trapped in a local minimum or a shallow valley when a much deeper minimum is nearby. The most common error function used to traverse the surface during backpropagation is the global measure of network energy called MSE. MSE is defined as the mean square error between the actual network outputs and the desired network outputs, the former being functions of a set of network parameters, or variables, namely the network weights (Pao, 1989; Wasserman, 1989).

Using MSE as the error function for single-hidden-layer networks having linear activation functions provides PCA of the input data (Oja, 1992). AANNs are often referred to as nonlinear PCA because of the use of nonlinear activation functions. A brief review of PCA follows. In the case of linear data, PCA is often used as a means to represent data in a reduced dimensional format (Baldi and Hornik, 1989; DeMers and Cottrell, 1993; Gonzalez and Woods, 1992; Hecht-Nielsen, 1990; Kramer, 1991;

Malthouse, 1996; Oja, 1992). Briefly, the principal axes (directions) of a region of pixels in an image are the eigenvectors of the covariance matrix of that region. The degree of spread around these axes is measured by the corresponding eigenvalues. Thus, the principal spread and direction of a region of data can be described by the largest eigenvalue and its corresponding eigenvector. Although multiple eigenvalues and eigenvectors exist for the input data, typically for images, only a few eigenvalues are very large and the remaining ones are very small. The smaller eigenvalues indicate that the regional components of the image contain limited information about the overall image. Data compression by PCA involves determining the largest eigenvalues and their corresponding eigenvectors and truncating the smaller valued eigenvalues and their eigenvectors. The image can then be represented in reduced dimension by its projection onto the subspace created by the truncated set of eigenvectors (Gonzalez and Woods, 1992).

Specifically, given an original image matrix $X$ and its set of eigenvectors, the subspace spanned by the eigenvectors associated with the largest $k$ eigenvalues will have a smaller mean square deviation from the original $X$ matrix than any other subspace with dimension $k$. For example, for a matrix $A$ of dimension *(p by k)*, where the columns are an arbitrary basis for a subspace in $\Re^k$, then mathematically

$$\min_A \left\| X - proj_A X \right\| = \left\| X - proj_{U_{(k)}} X \right\| \tag{11}$$

where $proj_A X$ and $proj_{U_{(k)}} X$ denote the projection of the rows of $X$ onto the subspace spanned by the columns of $A$ and by the $k$ eigenvectors (columns) corresponding to the first $k$ largest eigenvalues of $X$, respectively (Malthouse, 1996). In other words, representation of the data projected onto this subspace, created by the truncated set of

eigenvectors, will be the optimum representation of any subspace of the same dimension in assessing the mean-squared deviation from the original matrix. PCA, designed for linear data, often provides an over-dimensional linear fit to the data and does not provide the ability to reduce a nonlinear input data set to its intrinsic dimensionality in a nonlinear fashion (DeMers and Cottrell, 1993). AANNs, however, have this capability.

When the MSE error function is used for training an AANN, the process is similar to calculating the truncated set of eigenvectors, the significant difference being that the AANN contains nonlinear activation functions on the hidden layers and, therefore, introduces a nonlinearity into the calculation. Thus, AANNs are often referred to as nonlinear PCA of the data. In general, AANNs perform dimensional reduction nonlinearly, in much the same way that PCA performs this operation on linear systems.

*2.3.2.1.4 Architecture*

Determination of the network architecture is basically a trial-and-error process. The number of hidden-layer nodes is intricately related to the number of training samples and to the number of features per training sample. If the number of nodes in the hidden layers is insufficient to represent the function fully, the network "system" will be underdetermined, and the network will not approximate the underlying function. If the number of nodes is excessive, with an insufficient number of training samples, the network system will be overfit and will memorize the stochastic noise properties rather than generalize to the underlying signal. This can be demonstrated by examining the error on the training set and on the test set. If the error on the test set exceeds that on the training set by a factor of two or more, it is probable that the training set has overfit the network (Kramer, 1991).

The simplest approach to estimating the number of hidden-layer nodes is to restrict the number of weights in the network to a fraction of the number of constraints imposed by the input data. Estimating the maximum node limits of network architecture involves the use of information-theory concepts that are based on the number of training samples, *j*, and the number of features per training sample, *n*. The number of free parameters in an AANN must be less than $j \cdot n$ (Kramer, 1991). For the typical three-hidden-layer AANN architecture, the number of free parameters is

$$(n+m+1)(M_1+M_2)+n+m \tag{12}$$

where $M_1$, *m*, and $M_2$ represent the number of nodes in the first hidden layer, in the bottleneck, and in the third hidden layer, respectively. Therefore, according to Kramer (1991),

$$(n+m+1)(M_1+M_2)+n+m < j \cdot n \tag{13}$$

$$\rightarrow M_1+M_2 < \frac{j \cdot n - n \cdot m}{n+m+1} \tag{14}$$

$$\rightarrow M_1+M_2 < \frac{n(j-m)}{n+m+1} \tag{15}$$

Assuming *b*<<*f*, then Equation (15) is approximated by

$$\therefore \frac{n(j-small)}{n+1} \Rightarrow j \tag{16}$$

Therefore

$$M_1+M_2 << j \tag{17}$$

Thus, the sum of the number of nodes in the first and third hidden layers must be much less than the number of training samples for the data set. Additionally, $M_1$ and $M_2$ must each be greater than the number of nodes on the bottleneck, *m*, since the bottleneck by

design occurs in the second hidden layer of the combined network. These constraints on $M_1$ and $M_2$ provide initial guidance for implementing an AANN architecture.

The values of the projections on the bottleneck layer -- the compressed representation -- are dependent on the architecture of the network. Therefore, the projection vectors cannot be analyzed to obtain the values of the vector components since the projection vectors vary based on the network architecture. However, if two networks having different architectures produce the same results, then a homeomorphism exists between the networks, according to the theorem by Malthouse (1996). In summary, for an AANN, any network architecture that provides converged autoassociative output is satisfactory.

*2.3.2.1.5 Activation Functions*

Multilayer networks have greater representational power than single-layer networks when a nonlinearity is introduced. The activation function on the hidden layers is used to produce the needed nonlinearity. The choice of activation function is generally not critical. If the backpropagation algorithm is chosen, the only requirement is that the function be everywhere differentiable, and for universal approximation the function must be bounded (Cybenko, 1989). The sigmoid satisfies these requirements, as do the hyperbolic tangent and the linear functions. The sigmoid and the hyperbolic tangent are sometimes called logistic or squashing functions; they compress the range of the node output to values between 0 and 1 and between –1 and +1, respectively. The additional advantage of these logistic functions is that they provide a form of automatic gain control. For small signals the slope of the input/output curve is steep, producing high gain. As the magnitude of the signal increases, the gain decreases. In this way large

signals can be accommodated by the network without saturation, while small signals are allowed to pass through without excessive attenuation.

*2.3.2.1.6 Stability*

One means of determining the performance of an AANN is to validate its operation in the range over which it was trained. If the AANN interpolates data points based on the training data set, rather than extrapolating outside the range of the training data set, it is considered to be stable (Reimann, 1998).

As described previously, artificial neural networks are known to be universal approximators and provide a representation of a general function of an n-dimensional real variable, $x \in \Re^n$, by a finite linear combination of the form [Equation (5)]

$$\sum_{j=1}^{N} \alpha_j \cdot \sigma\left(y_j^T \cdot x + \theta_j\right)$$

where $y_j \in \Re^n$ and $\alpha_j, \theta \in \Re$ are fixed and $\sigma$ is the activation function of each of the nodes of the neural network (Cybenko, 1989). The AANN previously shown in Figure 10 can be viewed as two single-hidden-layer networks -- the encoding network and the decoding network -- cascaded together. The work of Cybenko demonstrated that both mappings $f$ and $g$ provide universal approximating power $x \xrightarrow{f} \alpha \xrightarrow{g} x$, with $f$ mapping the input to a reduced space and $g$ mapping the reduced space to target values. Therefore, AANNs are mechanisms for mapping data in the space of continuous functions. From this knowledge, a mathematical metric can be used to evaluate the stability of the AANN.

Mathematically, the stability of an AANN is determined by the infinity metric: $dist_{\infty}(x,z) = \max\{|x_i - z_i| : i \in target\ values\}$, that is, the maximum distance between the target and the output values in the fully trained AANN. Specifically, for $x$ (the target output value) and $z$ (the actual network output), where $z$ is regarded as a weak perturbation of $x$, the AANN must be constructed such that

$$dist(x,z) < \varepsilon \qquad \text{where } \varepsilon > 0 \tag{18}$$

An AANN constructed with every point $z$ being locally asymptotically stable is operating in the range of the training data and is interpolating -- not extrapolating -- information (Reimann, 1998). The current research demonstrates a method of validating the operation of an AANN by evaluating the difference between the target values and network outputs for each feature of each training sample. An evaluation of the differences, the ε values, determines the stability of the AANN. Difference values greater than ε demonstrate that the AANN is operating outside the stability range. This stability criterion is ultimately a means of assessing the generalization robustness of the AANN and a quality metric for evaluating the testing capability of the network -- whether the AANN is interpolating or extrapolating during its performance with testing data.

*2.3.2.1.7 Conclusions*

AANNs have been analyzed extensively for image and data compression by numerous researchers and are typically used to process same sensor information. AANNs are called nonlinear PCA when the MSE error function is used in backpropagation. Since the goal of the present research has been to correlate images from two sources, the extension of AANNs to process two different data sets is discussed in the next section.

#### 2.3.2.2 *Heteroassociative Neural Networks*

In the previous section the capabilities of AANNs and the limitations involved in implementing them were described. Issues related to error- and activation-function selection and architecture development are the same for Heteroassociative Neural Networks (HANNs), which involve training the input vectors to a different output vector -- training Signal 1 as input to Signal 2 as output. For HANNs, the same network as shown in Figure 9 is used, except that the training is to an output that is different from the input. Using the error function of MSE, Signal 1 is reduced to the space on the bottleneck. The first half of the network produces this reduction and generates a truncated set of eigenvectors for that signal, and the bottleneck is the projection onto that specific eigenspace. The second half of the network uses the projections (nonlinear) to find an appropriate space for restoring the desired output image. Effectively, two separate (nonlinear) eigenvalue problems are being solved, with the only commonality being the projection on the reduced space. From these networks, however, the relationship between the signals is neither clear nor obvious; the network incorporates the relationships, and no real connection can be interpreted from the results.

The HANN, as a specific architecture design, came about as a direct extension of the AANNs whose primary initial motivation was data compression (Cottrell and others, 1989; DeMers and Cottrell, 1993; Hecht-Nielsen, 1990 and 1995; Padgett and others, 1998; Turk and Pentland, 1991). The HANNs, training an input to an output, is a specific case of the generic multi-layer perceptron, as described in Section 2.3.2. This network configuration is referred to as a HANN -- instead of a multi-layer perceptron -- when a compressed representation of the input data is needed for further use.

Another type of HANN, shown in Figure 12, provides reduction of two variables to the reduced bottleneck. The encoding portion reduces both data sets to a combined intrinsic dimensionality. The second and third hidden layers and the output layers are considered to be the decoding portion of the HANN. This architecture can provide additional relationships within the features of two independent signals (Wasserman, 1989).

The network shown in Figure 12 is capable of learning not only the auto-associative relationship, i.e., learning input to same output, but also the interrelationships of the two images. This provides the ability to find the functional response of the mutual data set. This type of network does find a reduced dimensional representation of both data sets. However, the output, containing both signals, finds a mutual-space representation of the signals of interest.

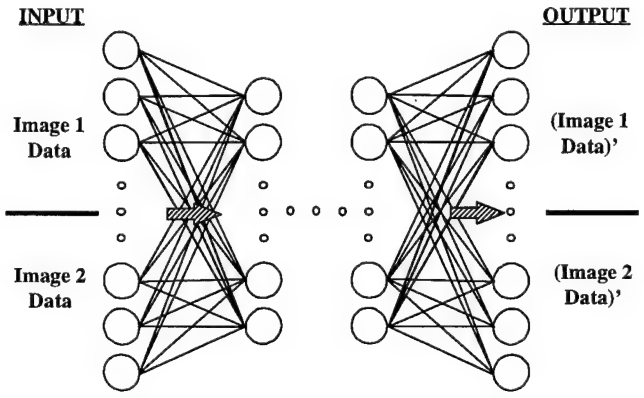



**Figure 12. Joint-Data HANN (Wasserman, 1989).**

*2.3.2.2.1 Conclusions*

HANNs can associate data from two different sources, and the interrelationships of the data are contained within the network. However, analysis of the correlation of the

features from the two sources is difficult and HANNs do not provide the ability to predict or synthesize one set of image data from another.

## *2.4 Conclusions*

Correlating data from different sensors is a multi-step process. The pertinent features must be extracted and an algorithm used to correlate them. Data fusion typically involves determining the features that represent the same material and developing methods of isolating those features in each source. However, this does not provide a general solution. Often the data sets require one specific set of algorithms for extracting the features and another for relating those features. What is needed is an effective method of finding features and a single algorithm for correlating them.

In image analysis, the concepts of the human visual system can be very helpful in extracting features. The next chapter details research that employs the concepts of the human visual system to extract features with the aid of Fourier and wavelet transforms. A new interpretation of AANNs is presented, i.e., the concept of using AANNs as filters. Although the architecture of Figure 9 in the HANN configuration predicts one sensor from another, it provides no means of ensuring the reliability of this prediction. Therefore, a new type of neural network architecture is introduced that predicts one sensor from another and generates an autoassociative output which can be analyzed for stability to validate the overall network operation.

# 3 *Research Approach/Methodology*

## *3.1 Introduction*

This chapter discusses the approach used in conducting the research on image analysis described in this dissertation. The research objective was to analyze two sensor images and predict or synthesize one image from another. The design principles of the human neural system relevant to feature selection from image data and the concepts of the human visual system served as an inspiration for determining the relevant data. Fourier and wavelet transforms were explored for their usefulness in selecting features to be used for predicting one sensor from another. For the synthesis of one image from another, the AANN was extensively analyzed and a new interpretation developed whereby the AANN can effectively be viewed as a data-filter device. From the AANN study a new neural-network architecture was developed as a means of correlating and predicting one sensor from another as well as generating an autoassociative output for validation of the overall network operation.

## *3.2 Implementation of Human-Visual-System Concepts*

The concepts of the human visual system were examined for their ability to find distinct features in two sensor images -- the key to the ability to predict one sensor from another using neural networks. Several human-visual-system concepts were described in detail in Chapter 2. This system performs different types of transformations, the ones of interest to this effort being those that provide form, shape, or feature information on the data set. Primarily, the transformations examined were those that maintain the low-frequency components and the form/shape information of the data as opposed to complex

logarithmic spatial transformations based on the anatomy of the visual pathway. Specifically, the Fourier and wavelet transforms were investigated as a means of analyzing the image data for specific features related to forms, shapes, or features. The Visual-Difference Predictor (VDP), by Daly (1993), was introduced as a tool for assessing image quality; this algorithm produces a map of the differences in the individual pixels of two images. This pixel-by-pixel metric can be used as a means of training a neural network to process image data. In Appendix A, the VDP is proposed as an alternative to MSE for use as an error function.

### *3.2.1 Feature Extraction Using Fourier Analysis*

Fourier transformations provide a representation of input data that is non-localized. Thus, a spatially local feature in the original image is represented globally across the many frequencies it contains. Since these transformations do not isolate specific spatial features, they are not effective for feature extraction for this particular application. However, Fourier data can be processed by neural networks. When real-valued image data is Fourier transformed, the coefficients become complex-valued data. Although neural networks process real, single-valued data, they can process complex-valued data, provided the complex values are pre-processed. Appendix B describes specific options for neural-network processing of complex-valued data. In summary, for processing complex values through an AANN, the data must be pre-processed/formatted in such a way that all input values to the network look real and single valued; this is the only means of determining the proper direction of the gradient descent for the weight updates.

A series of windowed Fourier transformations called the Gabor expansion (Graps, 1995) was also examined for its ability to extract localized features. The Gabor filter, like the Fourier transformation, spreads the frequency information in one location of an image throughout the Gabor output and, therefore, does not isolate specific spatial features. These filters have the ability to provide some spatial localization that is not possible with the gross Fourier transformation, but additional loss of information occurs. The Gabor expansion generates spatial band-pass filters. As the band-pass window becomes smaller, fewer low-frequency components are present, but the time localization improves; conversely, as the windows become larger, time localization decreases and the frequency presentation improves. Each use of the band-pass filter causes some truncation of data, namely, the high-frequency terms of each bandpass. As a result, reconstruction back to the original data is very lossy. Thus, Fourier and Gabor transformations could not be used for feature extraction for the current effort since they result in very global characteristics and specific localized features are necessary for the comparative analysis of two sensor images.

### *3.2.2 Feature Extraction Using Wavelet Analysis*

Wavelets are mathematical functions that partition a data set to its different frequency components and then examine each component at the resolution that matches its scale. The key to wavelet analysis is determining the wavelet that appropriately analyzes the data set. Many wavelet functions have been developed (Graps, 1995); thus, a literature search was conducted to determine the wavelet function most suitable for processing image data. This search revealed that the Daubechies wavelet has been used extensively for this purpose (Graps, 1995; Hubbard, 1996). Therefore, the initial

investigation of wavelets for feature extraction involved examination of Daubechies wavelets.

The literature documents significant use of the Daubechies wavelet as an effective means of image compression (Graps, 1995; Hubbard, 1996). The mother wavelet is constructed such that it focuses on maintaining the low-frequency terms of the image data while providing the ability to incorporate sufficient high-frequency information to reconstruct the image data from a reduced set of wavelet coefficients, namely, the low-pass wavelet output. As discussed in Chapter 2, maintaining the low-frequency terms ensures that the overall form or shape of the data is preserved and that the low-pass-wavelet output contains most of the information content of the image. For image-compression applications, this means that the high-pass-wavelet outputs have very low energy and can be discarded without significantly affecting the visual quality of the image reconstructed from the low-pass wavelet output only.

Chapter 2 discussed the three considerations involved in selecting a wavelet -- the compact support, the number of vanishing moments, and the filter length. The Daubechies family of wavelets is finite, with compact support; thus, the wavelet has a value of zero outside a certain interval. A wavelet from this family can have any filter length since the Daubechies wavelets are generated iteratively, not analytically. The number of vanishing moments is directly related to the filter length and is an additional mathematical relationship that must be satisfied (Graps, 1995; Hubbard, 1996).

The Daubechies $W_4$ wavelet that is four coefficients in length and has two vanishing moments was selected for a variety of reasons. First, only two vanishing moments are necessary for image processing; more than two increases the computation

time but produces no gain in quality (Hubbard, 1996). Application of this wavelet, with its compact support, generates high-frequency wavelet-decomposition images having very low energy. This satisfies the requirement that the low-pass-filtered image contain maximum information, which is necessary for reconstructing the original image with minimal loss. In addition, this wavelet is sufficiently short that the edge effects are minimal after the circular convolution is performed during wavelet decomposition.

Wavelet decomposition via the Daubechies $W_4$ wavelet provides four subsampled, filtered images, as described previously in Section 2.3.1.2. The image of interest is the low-pass frequency image that effectively smoothes the data. For the CT and MR image data used in this effort, the low-pass wavelet decomposition provides discrimination of two distinct features -- the bone and soft tissue -- in each of the two sensor images. It is this discrimination that makes this wavelet an effective means of extracting features from both sets of sensor data being examined. The specifics of feature extraction from the CT and MR image data will be discussed in Chapter 4.

### *3.2.3 Conclusions*

Examination of the human-visual-system inspired the use of Fourier, Gabor, and wavelet transformations for predicting one sensor from another. The first two transformations are very global in nature and do not provide spatially local features; therefore, these transformations are not effective as feature extractors for this application, since local distinct features are necessary for correlating the images.

An initial examination of the Daubechies family of wavelets indicated that extraction of localized features from two images of different modality was possible. Use of these wavelets provided a means of downselecting to a specific wavelet for this

application. The short-length Daubechies $W_4$ wavelet provides not only compact support with two vanishing moments but also a means of extracting two distinct features from the CT and MR image data sets. The experimental results are presented in Chapter 4.

### *3.3 Filtering Aspects of the AANN*

Research on AANNs has been extensive and inclusive, as detailed in Chapter 2. The AANN is recognized as a device for mapping the input data from its ambient dimension to a reduced dimensional space, which, ideally, is the intrinsic dimension of the data[1]. Another means of evaluating data processing through an AANN involves the concept of filtering. The output of the AANN can be viewed as a network response to features that are selected and processed; the AANN filters the feature data and provides the filter response based on feature associations.

The AANN behaves as a filter because it is trained to respond to specific input/ output combinations. The data-filtering capability and the type of filtering performed by the AANN are based on the type of features used to train the network. For broad-based features, the AANN can only generalize to the full spectrum of information in the training features. For specific features, designed for specific generalizations, AANNs will generalize to common features among the training-set data. In filter language, the spectrum of the training data is the frequency response of the AANN filter. The AANN will "ring" when an input is presented that incorporates the same spectrum as that of the training data. If the AANN filter response covers a broad spectrum, then the likelihood

[1] Determination of the intrinsic dimensionality of the data is not always obvious, and is still an area of active investigation. For linear data, PCA can provide the intrinsic dimensionality of the data, but for nonlinear data, other methods must be used (Kirby and Miranda, 1994 and 1998) and are still under investigation (1998 AFOSR proposal by Kirby, Miranda and Oxley). For this effort, trial and error, as discussed in Appendix C, was used to determine a satisfactory network architecture.

of ringing is very high when any input is presented. If the filter is highly focused to a specific range of responses, then the AANN will ring only for the response within the specific range for which it has been trained.

AANNs can be trained as very broad-spectrum filters. When image data are used for training samples that include the entire spectrum of information, the AANN trains to the entire image and effectively becomes a broad-spectrum filter for that particular image. Specifically, use of an entire image for each training sample creates an AANN having a very broad frequency response. The AANN learns to generate output on all the spectral components present in the training data. When training on entire images, the AANN responds to any input that contains the same spectral components as the entire training data set. Entire images incorporate a wide range of spectral components. The AANN learns to output a specific response when the input data include the spectral components upon which it was trained. Thus, the AANN becomes a broad-spectrum image filter. When this filter is rung, the output is the spectral components upon which it was trained, namely, the training image; regardless of input, the output will be the representation of the training image sets.

Figure 13 is an image of a 48 x 48 pixel three-gray-level cross. Thirty-one similar cross images were generated, with each image having different resolution and noise properties. The 48 x 48-pixel matrices were re-formatted to 2304 x 1 column vectors. The 31 different images were used to train an AANN implemented as a three-hidden-layer neural network. Through trial and error, a satisfactory AANN architecture was chosen (see Appendix C for a discussion of specific issues concerning trial and error feature and architecture selection). The first hidden layer contains 15 nodes with a

sigmoid activation function; the second contains four nodes with a linear activation function; and the third contains 15 nodes with a sigmoid activation function. This AANN was trained on the 31 training samples in only two training epochs.

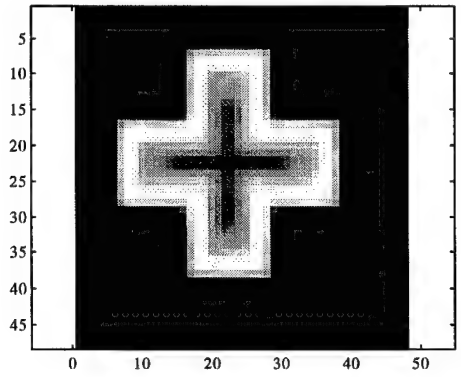



**Figure 13. Three-Gray-Level Cross Image.**

Figure 14 shows the output of the network when a cross image is applied to the input. The AANN has effectively and efficiently learned the three-gray-level cross image and is now tuned to the specific output on which it was trained -- the cross image.

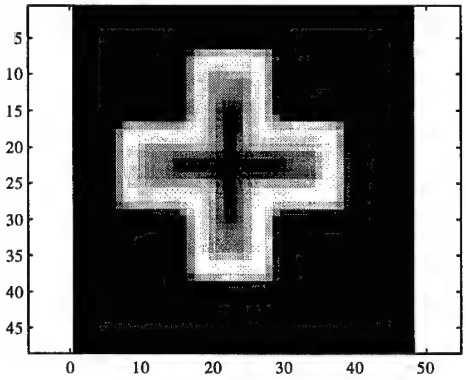



**Figure 14. AANN Output with Cross Image on Input.**

This is demonstrated in Figure 15, which is the output of the network when a 2304 x 1 column of zeros is applied to the input of the AANN. Figures 14 and 15 are

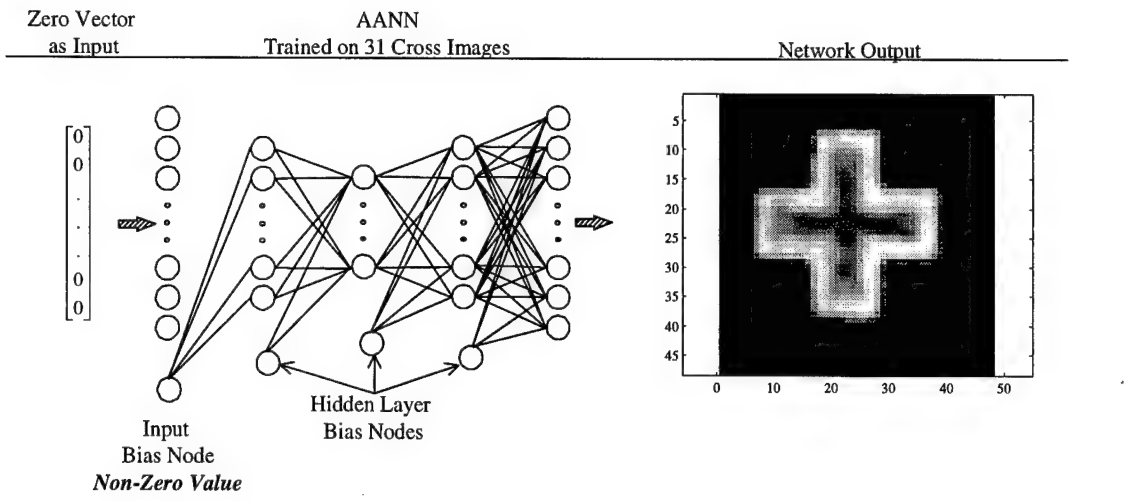



**Figure 15. AANN Output with Zero Vector on Input.**

very similar, and the mean-squared difference between the two images is 0.0026. Figure 15 shows that the AANN output is the image on which the network was trained, even when the input is zero. Examination of the filtering aspects of AANNs reveals that these networks learn a specific output response based on the input on which they are trained. For an AANN trained using broad-spectrum features, the output is the feature on which it was trained—in this case, the three-gray-level cross image. Each training sample incorporates all the spectral components of the image. The AANN has learned a specific output, namely, the three-gray-level cross image, based on the input. This AANN is a filter that responds with a three-gray-level cross output when the filter is rung on the input. Figure 15 shows that the AANN filter is activated by any input, even zeros, because it has learned to respond to a very broad set of spectral components. Further, the output response is primarily in the decoding portion of the network.

Figure 16 is the output when only the decoding layer of the network is calculated; the bottleneck node values are set to zero and only the bottleneck bias node is maintained. The AANN decoding portion of the network has learned the specified output from the

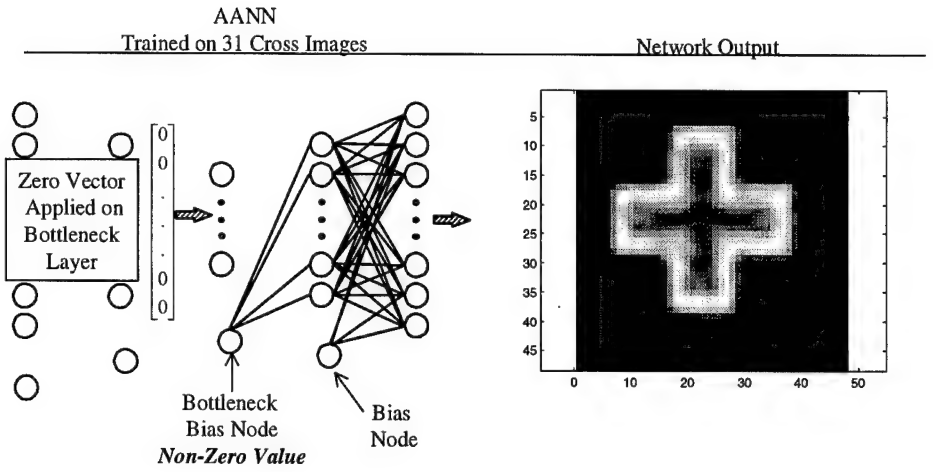



**Figure 16. AANN Output of Decoding Layer Only.**

input upon which it was trained. The output response is distributed through the decoding layers, including the bias nodes of the network. Figure 16 shows that the feature information is fully represented in the decoding portion of the network, where the only non-zero value on the bottleneck is the bias node. This node provides sufficient information to the decoding portion for generating the learned network output. For this example, the features are the images that incorporate all the spectral components representing this specific form or shape and that are maintained by the AANN. The AANN, trained with broad-spectrum data, is a broad-spectrum filter -- a filter that will output the image upon which it was trained, regardless of input.

AANNs can also be trained as limited, highly focused filters. In their typical use for image compression, they are trained to process gray-scale images and compress the data on the middle hidden layer (Cottrell and others, 1989; DeMers and Cottrell, 1993; Hecht-Nielsen, 1990). Fundamentally, the AANN learns the interrelationships of the gray-scale pixel values within the image(s) presented for training. This network is a

highly focused filter for gray-scale relationships in images and can be referred to as a gray-scale detector for compressing gray-scale images.

To create a generalized gray-scale-detector AANN, the training samples must represent the details of the image and also overlap sufficiently that the network will learn the commonality among all the samples, namely, the gray-scale relationships. For accomplishing this, a series of subimages, or tiles, from one original image is employed. Figure 17 illustrates the concept of subimage tiling. Each subimage tile represents one training sample to the AANN. The tiling window is moved across and down the image to generate a series of training samples that will represent the image being processed.

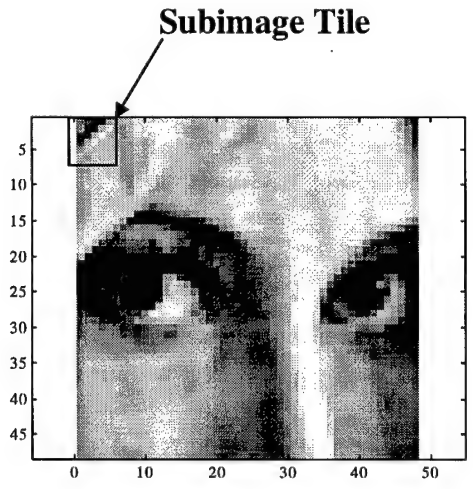



**Figure 17. Example of Subimage Tiling: "Lenna".**

The size of the tile is determined by the size of the image matrix being processed, by the resolution of the image data, or by trial and error. For many image-processing applications, a tile size of 8 x 8 pixels is selected (Malthouse, 1996). However, for small image matrices, this tile size incorporates a subimage that is too large to capture the detail in the image; therefore, smaller tiles must be defined. These smaller subimage tiles are used to train the network to a highly focused, small-response range. With a succession of

these subimage tiles, the AANN will find a solution that represents the commonality of the different subimage tiles. What the individual areas of the image have in common is the gray-scale relationship among the pixels. The resulting AANN has a highly focused, limited response range -- that of gray-scale detection. The AANN is now capable of processing and compressing a gray-scale image and reconstructing back to the original data set.

For the 48 x 48 pixel image of Lenna in Figure 17, 2209 training samples were generated by applying a 2 x 2 window, overlapping, across and down the image. The 2209 subimage tiles were used to train an AANN implemented as a three-hidden-layer neural network. Through trial and error, a satisfactory AANN architecture was chosen (see Appendix C for a discussion of specific issues concerning trial and error feature and architecture selection). The first hidden layer contains 15 nodes with a sigmoid activation function; the second contains four nodes with a linear activation function; and the third contains 15 nodes with a sigmoid activation function. Figure 18 shows network outputs of the trained AANN tested with two different gray-scale images. Figure 18, left, is the output of the AANN tested with the subimage tiles of an added-noise version of the Lenna image; the image on the right is the output of the AANN tested with subimage tiles of a cross image. This figure demonstrates that the network as trained can process any gray-scale image, and the AANN architecture provides a compression and reconstruction capability. The AANN has learned the very specific interrelationships among the training samples. In this case, the common bond is the gray-scale relationship within the image. The AANN, trained with highly focused data, i.e., a set of features that incorporate or are tuned to represent only a limited relationship between the pixels, has

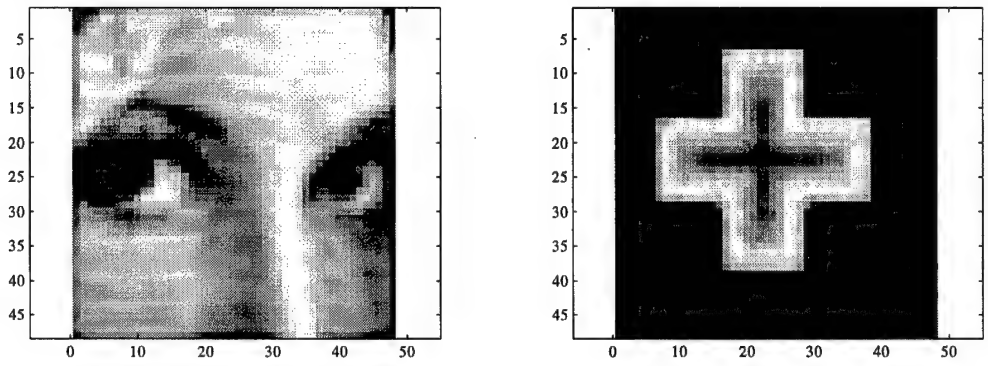



**Figure 18. AANN Output: Lenna on Input, Left; Cross on Input, Right.**

learned only the underlying relationship in the data. This set of features results in an AANN that is a highly focused filter -- a filter that is a gray-scale detector and can take in any gray-scale image and autoassociatively output the same image.

Whether the AANN responds as a broad-spectrum filter or as a highly focused filter is based on the selection of the features. The AANN trains to the underlying relationships that exist within the training data. Broad-spectrum features train an AANN to respond to all spectral components specified in the training samples. Highly focused features train an AANN to respond to the underlying commonality among the training samples. The selection of training features results in substantially different AANNs -- ranging from a very broad-spectrum response to a highly focused, limited response. Careful feature selection for AANNs ensures that the network will perform as desired.

### *3.3.1 Conclusions*

In the Fourier domain, it is quite natural to analyze AANN results as (nonlinear) filter responses. The selection of features determines the generalization and mapping that occur in the AANN in much the same way as AANN filter responses are driven by the training data.

### *3.4 Autoassociative-Heteroassociative Neural Networks*

Heteroassociative neural networks (HANN), as described in Chapter 2, provide a means of processing two image data sets simultaneously. In the first HANN described, when image data from one sensor on the input are mapped to those from a second sensor (as shown in Figure 9), the second sensor image can effectively be generated from the first sensor image. However, the relationship between the two image data sets is not an observable one. The network incorporates all the interrelationships, and the correlation of the features within the network cannot be directly interpreted.

The second type of HANN discussed was the joint HANN (shown in Figure 12), with both sensor image data sets functioning as input and output. This type of HANN provides a mutual mapping of the two image data sets that incorporates the interrelationships of the two images. However, the joint HANN does not provide the precise capability required to predict an image from a sensor, based on the image data from another sensor. Both image data sets must be input to generate an accurate prediction of one from the other; since the encoding portion of this HANN creates the joint basis space on the compressed representation, an input of zero as the second sensor input would not provide accurate prediction results on the output.

Neither of these HANNs enables the goal of this research to be met; therefore, a new type of neural network architecture is being proposed here -- the Autoassociative-Heteroassociative Neural Network (A-HNN), as shown in Figure 19. The A-HNN provides a prediction capability and an autoassociative output. This network maps from one input image to itself and to a different output image, and the middle hidden layer represents the projection that is used to map from the original image to itself,

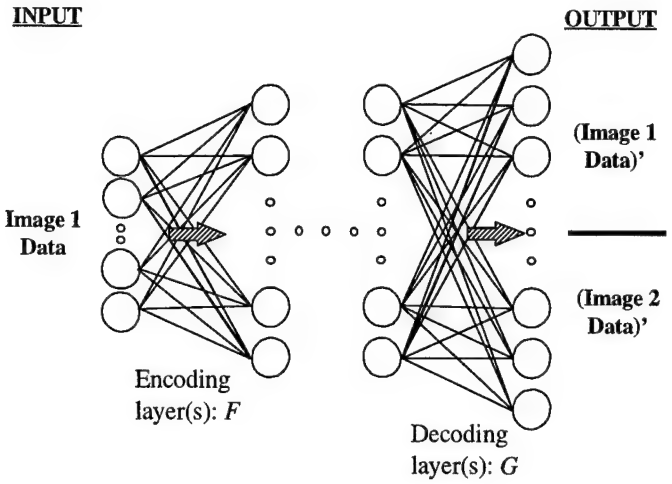



**Figure 19. Autoassociative-Heteroassociative Neural Network (A-HNN).**

autoassociatively, as well as to the second sensor image. This network will provide an output that predicts the second sensor image based on the reduced dimensional-projection information that is present only in the original image.

The operation of the A-HNN parallels that of the AANN and the HANN as a data-mapping network. In a specific three-hidden-layer implementation, the data are mapped from the input to a reduced space and to an output. The A-HNN, like the AANN and the HANN, is a data-mapping device; however, it provides a mapping from one variable to two variables. On the decoding layer, the network finds the joint function, *G*, between the images, based only on the information content in the original image reduced on the encoding layer, *F*.

Though this new architecture parallels the operations of the AANN and the HANN, the A-HNN significantly departs from the operations of the "typical" AANN or the joint HANN. The joint HANN provides an *implicit* identity mapping of two vectors, $(I_1, I_2)$ on input to the same vectors on output, $P(I_1, I_2) = (I_1, I_2)$. The two-vector output is created from the input vectors operated on by *P*, a composite function created from the

encoding layer function, $A$, and decoding layer function, $B$, of the network: $P = B \circ A$, as the mapping function. Concatenation of these two vectors provides the ability to analyze them separately; the individual vectors operated upon by the network are defined by $P = (P_1, P_2)$ such that $P_1(I_1, I_2) = I_1$ and $P_2(I_1, I_2) = I_2$. Defining the decoding layer $B = (B_1, B_2)$ such that $B_1(\textit{middle layer nodes}) = I_1$ and $B_2(\textit{middle layer nodes}) = I_2$, the composite function of the network becomes $P = B \circ A = (B_1 \circ A, B_2 \circ A)$ such that $P_1 = B_1 \circ A$ and $P_2 = B_2 \circ A$. The new A-HNN architecture provides the *inverse* of the encoding layer and the individual $B$ decoding layers of this AANN, that is, $Q = (Q_1, Q_2)$ where $Q_1(I_1) = (I_1, I_2)$ for $Q_1 = P_1^{-1} = A^{-1} \circ B_1^{-1}$ and $Q_2(I_2) = (I_1, I_2)$ for $Q_2 = P_2^{-1} = A^{-1} \circ B_2^{-1}$, providing the inverse of $A$ exists[2]. Therefore, the A-HNN encoding layer, $F$, is equivalent to $B_1^{-1}$ or $B_2^{-1}$, and the decoding layer, $G$, is equivalent to $A^{-1}$, regardless of which image is processed on the input.

Figure 20 is a specific implementation of the A-HNN, a three-hidden-layer architecture, paralleling the architectures described for the AANNs and HANNs. The input and the first and second hidden layers constitute the encoding portion of the network. With input data from one source, $x$, where $x \subset \Re^{n_1}$, the A-HNN is used to map the data set to a reduced dimension or projection space, $\alpha$, i.e., $F : \Re^{n_1} \rightarrow \Re^m$, where $\alpha \subset \Re^m$ and $n_1 > m$. The function $F$ approximates a mapping from the original data set to its intrinsic dimensionality, $m$.

[2] $A^{-1}$ exists if and only if the network maps to the intrinsic dimensionality of the data which provides a 1-1 and onto mapping. Higher dimensional mappings result in some redundancy, and will not be 1-1, but still permit the existence of $A^{-1}$. If the network maps to a dimension lower than the intrinsic dimension, then $A^{-1}$ will not exist.

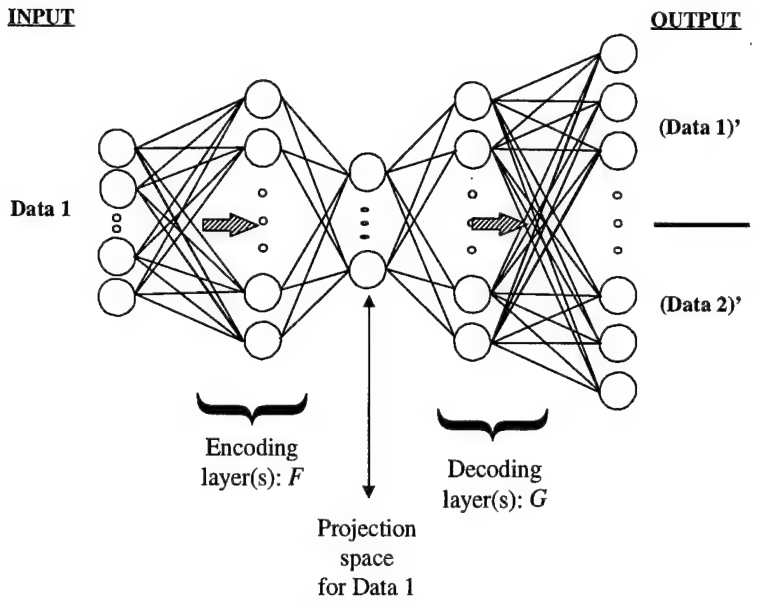



**Figure 20. Three-Hidden-Layer A-HNN.**

The second and third hidden layers and the output constitute the decoding portion of the A-HNN. Here the middle hidden layer, representing the projection space of the input data, is mapped to the original data and the data to be predicted: $G : \Re^m \rightarrow \Re^{n_1+n_2}$, where $y \subset \Re^{n_2}$. Therefore, the A-HNN provides a mapping $(x) \overset{F}{\rightarrow} \alpha \overset{G}{\rightarrow} (x, y)$ from the original input data to itself, autoassociatively, and also to a second sensor image for purposes of predicting the second sensor image from the first. The function $G$ determines the autoassociative relationship from both the reduced projection space $\alpha$ and the second sensor that is constrained by the projection space created from the first sensor. The function $G$ now represents the interrelationship of the two sets of sensor data and predicts the second sensor image from the first. The autoassociative and heteroassociative relationships in the A-HNN permit verification of the operation and overall testing quality of the network, and provide interrelationships of the sensor images respectively.

As described in Chapter 2, the stability of an AANN is validated by verifying on a feature-by-feature basis that the distance between the target and network output is asymptotically stable, which signifies that the AANN is performing within its training-data range. This stability criterion for AANNs can be extended to the autoassociative portion of this new A-HNN architecture, as a means of evaluating and verifying the operation and results of the entire network operation.

As shown in Figure 20, the encoding and decoding layers of the A-HNN are data-mapping devices that represent a set of functions that are dense in the space of continuous functions with respect to the maximum norm and, therefore, are universal approximators. As with the AANN, this new network architecture can be viewed as two single-hidden-layer networks -- the encoding network and the decoding network -- cascaded together. From the work of Cybenko (1989), both mappings $F$ and $G$ provide universal approximating power $(x) \xrightarrow{F} \alpha \xrightarrow{G} (x, y)$; $F$ maps from one input to a reduced space, and $G$ maps from the reduced space to the two variable target values. Like AANNs, the A-HNNs are mechanisms that map data in the space of continuous functions and, therefore, the stability metric of the AANNs can be utilized to evaluate the A-HNN performance.

The infinity metric, i.e., the maximum distance between the target values and the output values achieved by the autoassociative portion of the network, is used to evaluate the stability of the A-HNN. Specifically, for the target output value of the autoassociative portion, $x$, and the actual autoassociative network output, $z$, where $z$ is regarded as a weak perturbation of $x$, the A-HNN must be constructed such that [Equation (18)]

$$dist(x,z) < \varepsilon \qquad \text{where } \varepsilon > 0$$

For every A-HNN constructed, in which every point $z$ is locally asymptotically stable, the network is performing within the training-data range and is interpolating and not extrapolating information.

As in the case of the AANN, the stability metric is a means of validating the operation of an A-HNN by evaluating the distance between the target values and network outputs for each autoassociative feature for each training sample. An evaluation of the differences, the $\varepsilon$ values, determines the stability of the A-HNN. Differences above a chosen threshold demonstrate that the network is operating outside the stability range, e.g., outside the range upon which it was trained by the training-data set.

This research involves the processing of image data. Many problem applications for separating data into classes, or associating one set of data to another can be solved using neural networks. The A-HNN is fully capable of classifying data, and provides a network with improved training performance over a generic multi-layer perceptron. Appendix D addresses the use of the A-HNN for other than image data, and discusses the trade-offs of this new architecture.

### *3.4.1 Conclusions*

The A-HNN has the ability to produce one image from another. Inclusion of the autoassociative output enables validation of the total network operation as well as improvement of the training performance. The new network is a data-mapping device in which the encoding portion determines the functional approximation of the input data to a reduced dimensional space and applies constraints on the network. The decoding portion is the joint mapping from the reduced dimensional space to two image data sets. The

constraints applied from the encoding portion of the network ensure that the decoding transformation will be consistent with both the original data and the second data set.

### *3.5 Conclusion*

This chapter has discussed a theoretical investigation on the use of AANNs as filters and the development of a new network, the A-HNN, for predicting one sensor image from another. Concepts from the human visual system were examined for determining methods of extracting features from image data. It was found that wavelets would provide the means of extracting features from the CT and MR images under investigation. A new error function for training image-processing AANNs was proposed (see Appendix A). During investigation of the Fourier-transform space, a method of processing complex-valued data through AANNs was developed (see Appendix B).

The next chapter contains a discussion of the experimental results -- specifically, 1) feature extraction from the CT and MR image data using wavelet concepts, 2) implementation of the new A-HNN to predict a CT image from an associated MR image, 3) validation of the new network for operational robustness, based on the stability metric for the autoassociative portion, and 4) evaluation of the predictive output against the human-visual-system concepts discussed in Chapter 2, namely, Fourier distance and use of the VDP to determine similarity.

# 4 *Experimental Results*

## *4.1 Introduction*

This chapter discusses the experimental results obtained through the use of features from the wavelet decomposition of two images as input to the A-HNN. Wavelet decomposition allows determination of distinct, specific features from each of the images. The A-HNN provides a means of predicting one sensor image from another and of validating the operation of the network. The predictive output of this new network is evaluated with reference to the human-visual-system concepts, which were discussed in Chapter 2 -- Fourier distance and the VDP for determining similarity.

## *4.2 Feature Extraction from Wavelet Decompositions*

This research was performed using the small images of a CT and MR scan in Figure 21 -- 48 x 48 pixel images primarily showing the bone at the back of the human skull and some nearby soft tissue. These are subimages of full-head images provided by the National Institutes of Health, National Library of Medicine, Visible Human Project. Both the CT and the MR images were obtained from the same head at the same location, with the head being in the same orientation. For this application, registration is not an issue, which reduces this effort to a two-dimensional problem.

As discussed in Section 3.2.2, the Daubechies $W_4$ wavelet was selected for image processing (Graps, 1995; Hubbard, 1996). This wavelet has two vanishing moments and, as a result of its compact support, generates high-frequency wavelet-decomposition images having very low energy. This creates a low-pass-filtered image that contains the

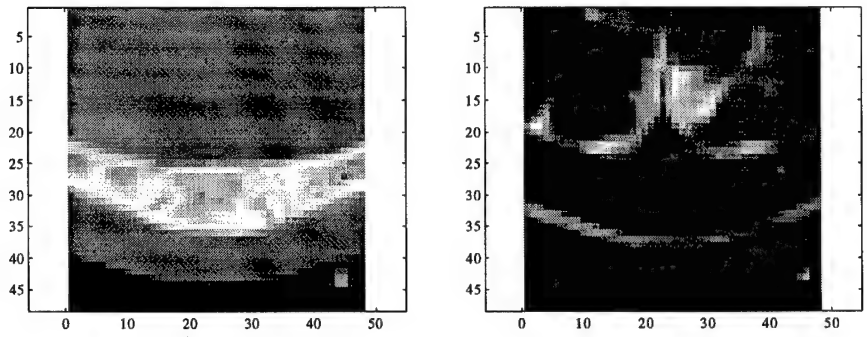



**Figure 21. CT Image, Left; MR Image, Right.**

information needed to reconstruct the original image, with a minimal amount of visual degradation. In addition, this wavelet is sufficiently short to minimize the edge effects from the circular convolution required for wavelet decomposition. For the 48 x 48 pixel images examined, longer filters provided undesirable edge effects.

This wavelet was implemented by a quadrature mirror filter, as discussed in Section 2.3.1.2, and created iteratively. Figure 22 shows the smoothing low-pass filter, *H*, and the detail high-pass filter, *G*.

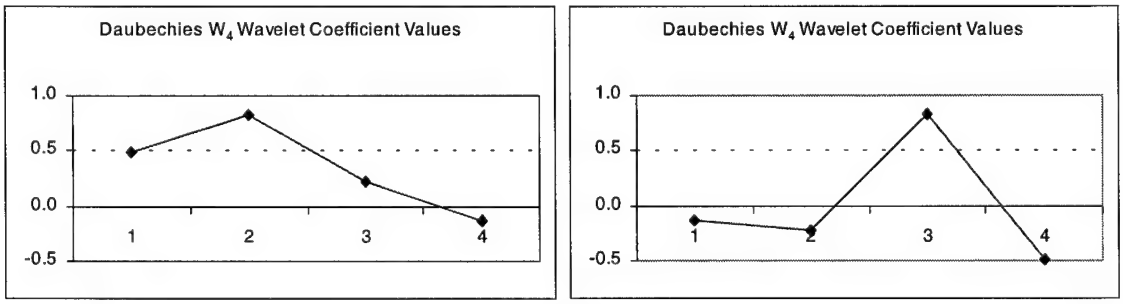



**Figure 22. Daubechies $W_4$ Wavelet: Low-Pass Filter *H*, Left; High-Pass Filter *G*, Right.**

After the wavelet transform was implemented, a single decomposition was performed, resulting in four images, $\text{Image}_{\text{LowLow}}$, $\text{Image}_{\text{LowHigh}}$, $\text{Image}_{\text{HighLow}}$, and $\text{Image}_{\text{HighHigh}}$. Each image was one-fourth the size of the original image, permitting all four images to be displayed simultaneously in a matrix the size of the original image.

The wavelet decompositions consist of four filtered images that are displayed in the quadrants of an image matrix as $Image_{LowLow}$ in the upper left, $Image_{LowHigh}$ in the upper right, $Image_{HighLow}$ in the lower left, and $Image_{HighHigh}$ in the lower right.

Figure 23 shows the four single wavelet-decomposition images for a CT image (left) and an MR image (right). In this figure the low-low-pass image (upper-left quadrant) provides the maximum amount of detail concerning the original image, although some loss from the original can be observed. Here, the high-frequency terms have been eliminated and, thus, the image is very effectively smoothed. The low-high-pass image (upper-right quadrant) provides the next level of detail, highlighting the dominant horizontal edges. The high-pass images (lower quadrants) provide a decreasing amount of visual information, with the high-low-pass image (lower-left quadrant) highlighting the dominant vertical edges in the images and the high-high-pass image (lower-right quadrant) providing very little visual information.

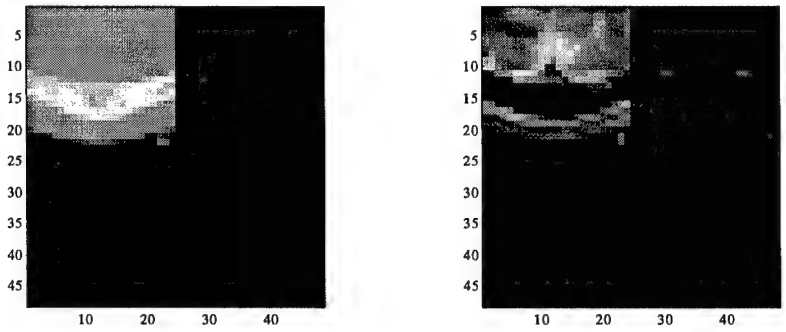



**Figure 23. Wavelet Decomposition: CT Image, Left; MR Image, Right.**

The low-high-pass (upper right) and the high-low-pass (lower left) images provide edge information; the dominant edge is associated with the skull bone. These "edge images" also contain substantial high-frequency components or noisy information associated with the edges, which makes analysis of these images very difficult. The

energies in all three of the high-frequency images tend toward zero, implying that very little information is present in the data. Therefore, these images are not very useful in distinguishing the bone from the soft-tissue features. Therefore, for this effort, the high-frequency, detail signal, images were disregarded, and only the low-low-frequency, approximation signal, image (upper left) was used for extrapolating features.

Figure 24 shows the low-low-pass wavelet-decomposition images for the original CT and MR images. A comparison of these two images provides a means of distinguishing the two specific features, i.e., bone and soft tissue, present in each image. The bone feature in these images is displayed as an arc in the bottom half of the image; since the thickness of the bone feature is relatively uniform throughout these images it is most accurately identified by column measurements. The black area at the bottom of the images represents locations devoid of material, e.g., where air is imaged around the head.

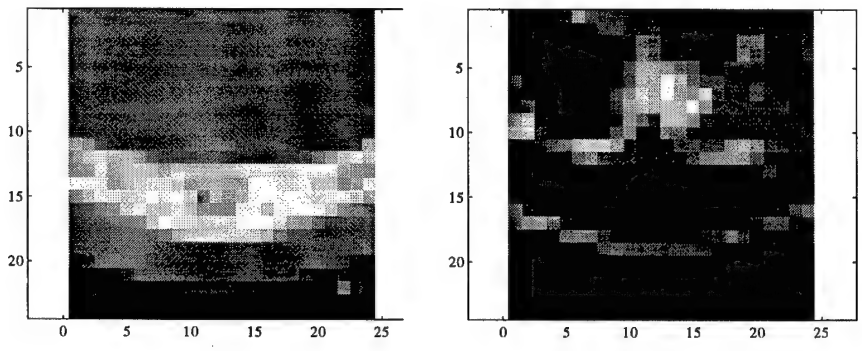



**Figure 24. Low-Pass Filtered Wavelet Decomposition; CT and MR Images.**

Figure 25 contains single-line plots of Column 10 of both the CT image (solid line) and of the MR image (dashed line) in Figure 24. These plots display the behavior of each image at a specific location. It is obvious that an approximate inverse symmetry

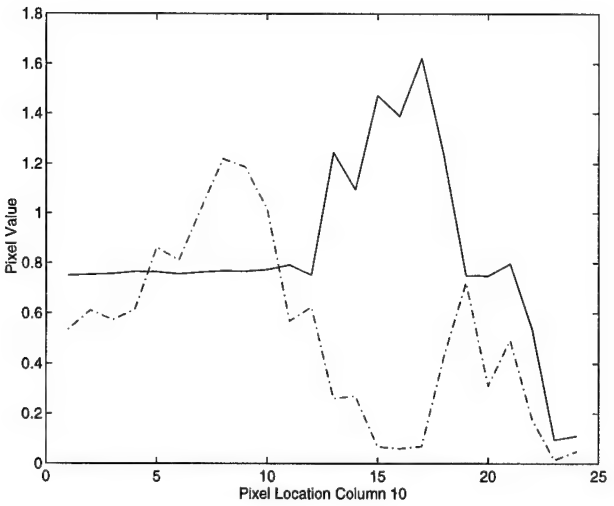



**Figure 25. Line Plots of Column 10 of Wavelet Decompositions: CT (solid) and MR (dashed).**

exists between the CT and the MR data at certain points (e.g., pixels 12-18). This symmetry can be observed because the low-pass filtering has effectively and consistently smoothed the data. Further analysis requires the use of *a priori* information concerning the images.

As described in Section 1.1, CT images reflect the density of the material under inspection, with high-density material being imaged as brighter pixels. It is well known that in an image, bone is indicated by the brightest pixels and the soft tissue, skin, and gray matter, by darker pixels. MR images basically reflect the amount of water present in the material; since soft tissue contains more water than bone, its image is composed of brighter pixels, and the image of bone is composed of darker pixels. Clearly, the CT and MR techniques discriminate between the soft tissue and the bone. In Figure 25, the peak on the CT plot and the valley on the MR plot are representations of the same feature, namely, the skull bone. Thus, two specific features can be extracted from each source -- one for bone and one for soft tissue. The features are composed of the data that represent bone (pixels 12-18) and soft tissue (pixels 1-11 and 19-23). It is noticeable that the

features are discriminated in blocks of six pixels. Because of the obvious separation present in the wavelet approximation image, a 6 x 1 window can be applied to generate the training samples for this experiment. This window must be applied in a non-overlapping fashion, to ensure that the bone and soft tissue features are segregated in the training data.

The orientation of the bone in the images in Figure 24 is primarily horizontal, which permits the use of this very simple method of discrimination. For these images, examination in two directions is not necessary, since the features are mainly in only one direction. Therefore, a 6 x 1 non-overlapping window is used to extract the bone and soft-tissue features from the low-pass-filtered images. This window provides a horizontal filtering of the approximation images, specific to these partial images; for complete head scans, a two-dimensional feature extraction would be required to incorporate the various angular orientations of the elliptically shaped skull bone.

As discussed in Section 3.3, neural networks operate as filters and ring, based on the training input data, to the specified output response. It is important to extract those features that discriminate the desired information. As shown in Section 3.3, a series of overlapping subimage tiles results in a neural network that is only a gray-scale detector and does not discriminate features within the gray scales. The features described above, obtained through wavelet decomposition, are the pixel data being used to discriminate between bone and soft tissue, rather than simply provide training on the different gray levels of the image.

## *4.3 Implementation of Autoassociative-Heteroassociative Neural Network*

Once the image features have been determined, the appropriate network architecture for processing these features must be chosen. The A-HNN, as shown in Figure 19, was selected to provide a prediction capability as well as an autoassociative output. The presence of both outputs provides a means for validating the network functionality.

Several issues required resolution prior to implementation of the A-HNN. The first involved determination of the architecture, including the number of hidden layers and the number of nodes per hidden layer. Section 2.3.2.1 discussed the various options for implementing a neural network. Although some guidance is available, trial and error is still the method of choice in determining a useful network architecture (see Appendix C for a discussion of specific issues concerning trial and error feature and architecture selection).

The first step involves the number of hidden layers. The discussions in Chapter 2 indicate that the typical configuration for AANN/HANNs is a three-hidden-layer architecture. This architecture was designed for data-compression applications in which the middle hidden layer is the compressed representation (Cottrell and others, 1989; Hecht-Nielsen, 1990 and 1995; Kramer, 1991). For the present study, compression was not an issue; therefore, for computational expediency, the first implementation of the A-HNN employed a single hidden layer.

The second step involves the number of nodes in the hidden layer. The only guidance on the appropriate number of nodes is that it should be less than the number of training samples, as discussed in Section 2.3.2.1.4. Use of a 6 x 1 non-overlapping

window on the low-pass wavelet-decomposition images resulted in 96 training samples per image, effectively separating the features of bone and soft tissue. The number of nodes is determined by trial and error, i.e., networks are run with different numbers of nodes to find the architecture which converges the weights most rapidly. Determination of "most rapidly" is very subjective; for this effort, the most rapid decrease in the MSE of the network output to the target values was used as the criterion for most rapid convergence of the network weights. Through successive trial-and-error network runs, it was determined that 15 hidden-layer nodes would produce the desired results.

The second issue to be resolved concerns the choice of nonlinear activation function. For this effort, the sigmoid activation function was chosen for the hidden-layer nodes, and the linear function was chosen for the output nodes. This represents a typical arrangement (Cybenko, 1989 and 1996; Kramer, 1991)[3].

The third issue involves choice of the training error function for backpropagation. Two specific error functions were considered: MSE and an alternative -- the VDP for use in training an AANN/HANN to learn image data. The VDP is designed to train the network to adjust the pixel-value outputs to the target values and, thus, learn the image data. This application, although it makes use of the image data, trains the A-HNN not only to the image but also to specific features within the image -- the bone and soft tissue. That is, along with the gray-scale relationship, the A-HNN must learn an additional relationship, namely, the difference between the bone and soft-tissue features. Since the VDP error function is designed to train to pixel values of the images, and is not designed

[3] The use of a logistic function, as discussed in Section 2.3.2.1.5, satisfies the requirements necessary for the network to provide universal approximation. Based on the concept of an AANN as a cascade of two single-hidden-layer networks, the middle-hidden layer can be viewed as the output of the encoding portion; output layers typically use linear activation functions unless gain control is required.

to incorporate the additional feature information present in these training samples -- the bone/soft-tissue information -- the MSE was chosen as the error function.

In the present study, the A-HNN was implemented as a single-hidden-layer network, with the architecture being six nodes on input, 15 nodes on the hidden layer with the sigmoid activation function, and 12 nodes on the output with the linear activation function. The training data consisted of the 96 vectors (6 x 1) of the MR image on the input that represented the top half of the target outputs and the 96 vectors (6 x 1) of the CT image that represented the bottom half of the target outputs. Figure 26 graphically shows the techniques used for feature extraction -- a non-overlapping 6 x 1 window applied to the low-pass wavelet-decomposed image. Figure 27 graphically depicts the order of data presentation for training the A-HNN -- the MR image features as the input and the MR and CT image features as the target outputs. This network implementation predicts the CT image data from the MR image data.

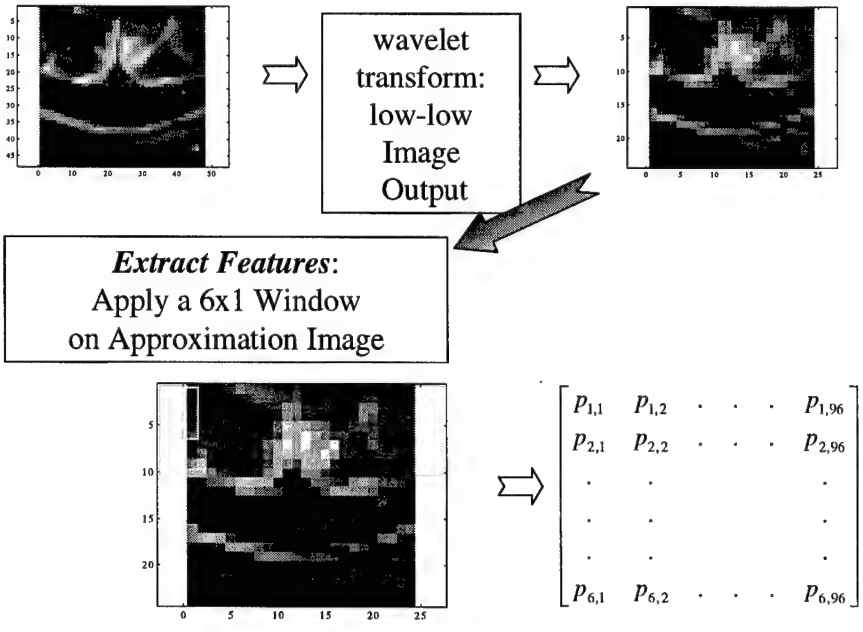



**Figure 26. Feature Extraction Technique.**

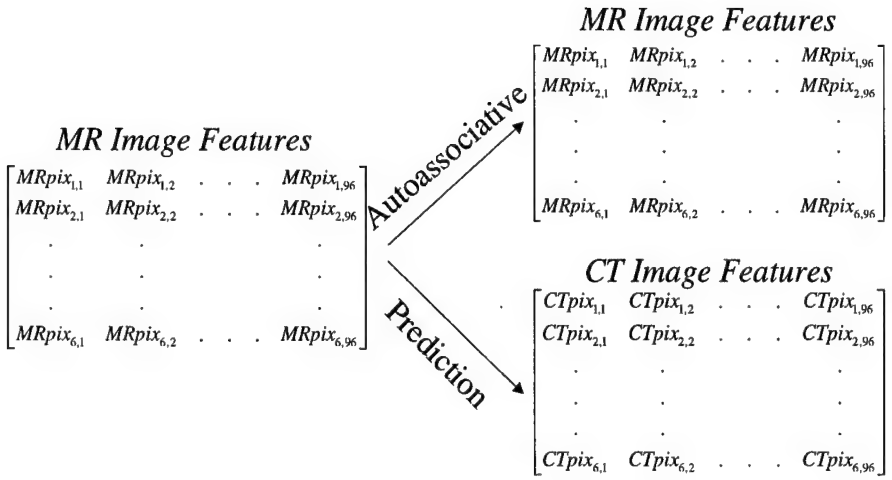



**Figure 27. Order of Data Presentation for Training A-HNN.**

Figure 28 shows the two network output images that resulted from an MR test data image applied to the A-HNN: MR image vectors (AANN portion of the network) (top) and the CT image vectors (predictor portion) (bottom). The MR test data image applied to the A-HNN was an added-noise version of the MR training image.

The network results appear to meet the subjective criteria for "similar". The first decision to be made is whether the network is operating in its stable region. This can be

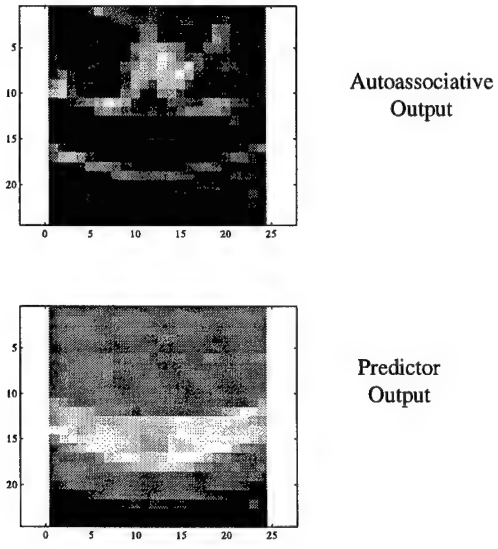



**Figure 28. A-HNN Output Images with MR Image on Input.**

determined, as discussed in Section 3.4, by examining the infinity norm of the autoassociative portion of the output, the infinity norm being the difference between the target value and the network output for each feature of the testing data samples. The A-HNN is stable if the autoassociative portion of the network output differs from the testing target values by less than a small value $\varepsilon$, where $\varepsilon > 0$, which indicates that the test samples are within the training range of the A-HNN. This analysis validates the performance of the A-HNN and demonstrates that the results of the prediction portion are accurate and real.

Figure 29 contain plots of the infinity norms of each feature in the testing data simulated through a trained A-HNN. Selection of $\varepsilon$ value is based on the problem requirements. For this problem, an $\varepsilon$ value of 0.1 is selected; since the data values have been normalized to 1.0, this $\varepsilon$ indicates at least 90% accuracy for each feature trained through the A-HNN. The upper and lower ranges of $\varepsilon = 0.1$ are shown by dotted lines in these plots. The test data consist of 96 vectors (6 x 1) obtained from a low-pass wavelet-filtered MR image similar to that used for training the A-HNN. Each plot contains a delta value (target value minus the network output value) for each of the 96 test samples simulated by the network. The plots show that the infinity norm of each feature is within the $\varepsilon$ tolerance; therefore, the A-HNN, as trained, is stable and the testing samples are within the response range of the network. Therefore, the entire A-HNN has been validated, and the prediction portion can be analyzed.

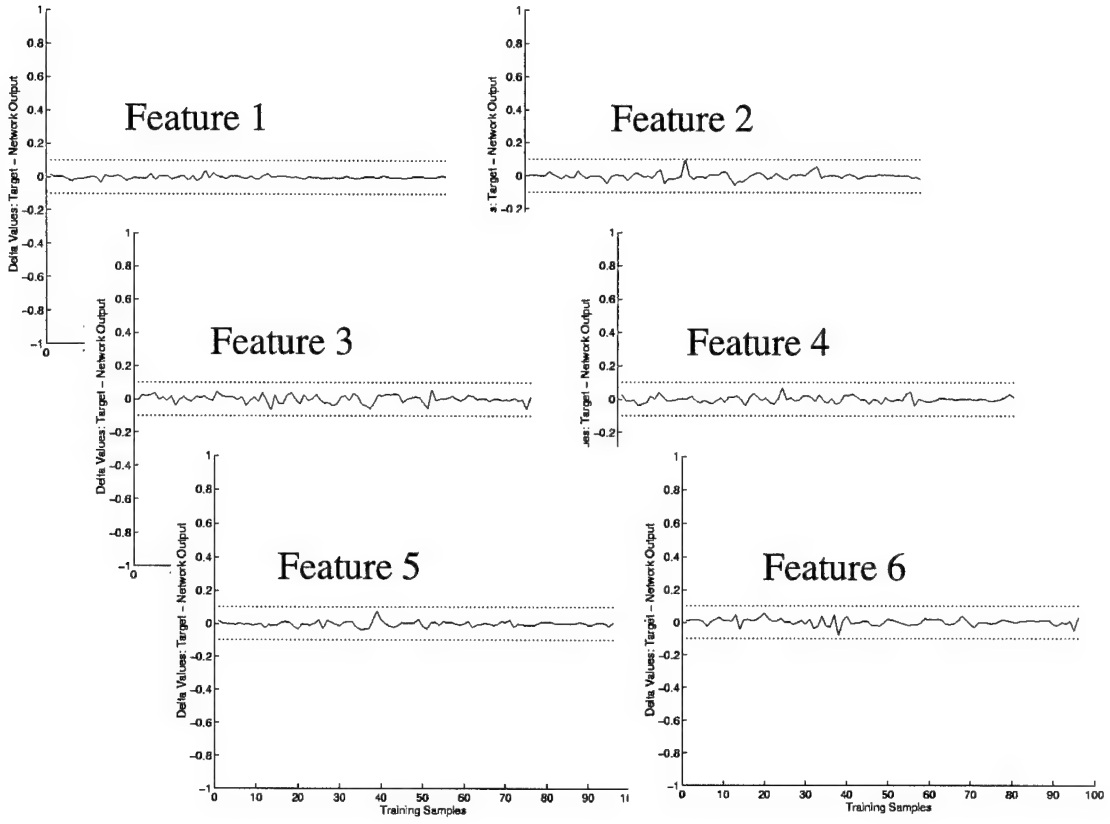



**Figure 29. Plots of Infinity Norms of Each Feature in the Testing Data.**

The next consideration concerns the accuracy of the images -- in other words, how similar are the images obtained by the network and the anticipated images. For evaluating the similarity of two images, three methods based on the human-visual-system concepts discussed in Chapter 2 were used: 1) a subjective human opinion on similar vs. dissimilar, 2) distance in Fourier space, where close in Fourier space indicates close by human-visual-system concepts, and 3) use of the VDP to reveal the pixels having variations greater than those that would be classified as similar by a human.

In the case of the first method, subjectively, the MR target image shown in Figure 24 (right) and the MR network output image in Figure 28 (top) have a very similar

appearance. The CT network output image in Figure 28 (bottom) likewise is very similar to the CT target output image in Figure 24 (left).

In the second method, transforming the images to Fourier space and taking the distances between the images produces the following results:

| **Images** | **Distances** |
|---|---|
| MR Target Image to MR Network Output | 0.0114 |
| CT Target Image to CT Network Output | 0.0195 |

**Table 2. Fourier Space Distance Between Target Values and Network Outputs.**

As stated earlier, when distance values are near zero, the images are of the same form or shape and belong to the same cluster of objects. Close in Fourier space implies similar by human standards. The definition of close is problem specific; for this application the data can obtain a maximum value of 2 which means both of these values are less than 1% variation from zero. Since the distance values are within 1% of zero, it can be concluded that the network outputs are similar to the target values.

In the third method, the VDP was used to compare the network output to the target images, and the results are shown in Figures 30 and 31. In Figure 30 the target MR image is on the top left; the network output image is on the bottom left; and the VDP output (probability of similarity map) is on the right. The VDP algorithm calculates "similar" pixels as zero or near zero and "dissimilar" pixels as near or at $\pm 1$. This algorithm incorporates a sign to identify the direction of the difference between the images. For the particular display software used, the values near –1 are displayed as black and those near +1 as white, and those near zero as gray.

Figure 30 shows the VDP output (right) created by comparing the MR target data (top left) and the network output (bottom left). In the VDP output the specific areas where the images are dissimilar are shown by black or white pixels. The gray represents those areas of the image which the human would describe as being the same or similar. Notice that the MR images are similar, except for the left-side edges of the bone. This indicates that the network has not accurately replicated the high-frequency components across the entire edge of the bone feature. The VDP highlights pixels that may be a potential problem. In this case, the other evaluations of the network output, i.e., subjective evaluation and the distance in Fourier space, indicate that these individual variable pixels are not sufficient for the human to judge the images as dissimilar. The location of the pixel variations as well as the frequency of the variations is not substantial enough to cause a ruling of dissimilarity.

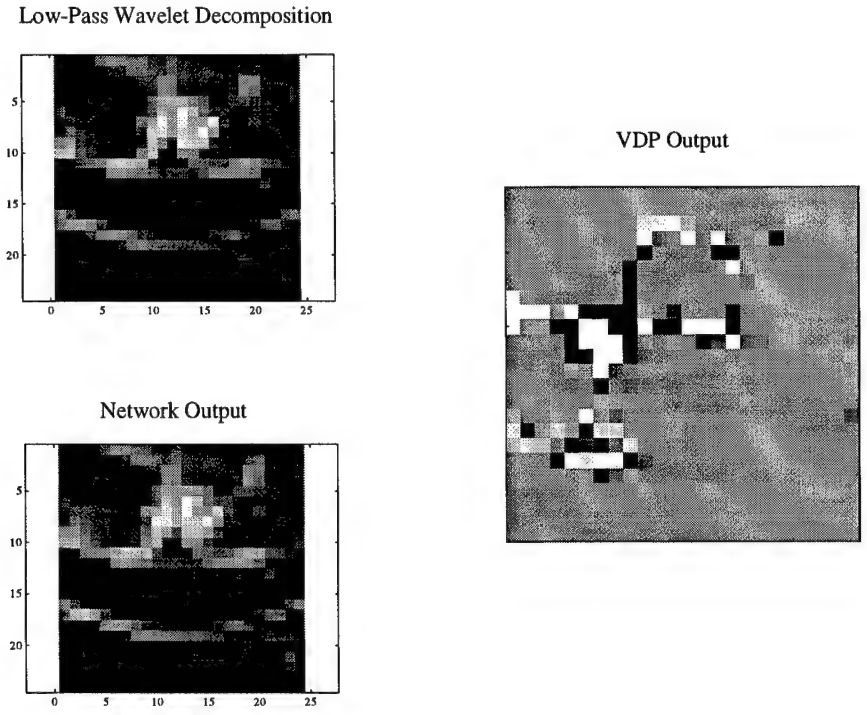



**Figure 30. VDP of A-HNN; MR data.**

Figure 31 shows the target CT image on the top left, the network output image on the bottom left, and the VDP output on the right. Once again, the VDP shows that the CT images are similar, except for the edges of the bone, indicating that the network has not accurately replicated the very high-frequency components at the edges. The white and black pixels in the upper right corner of the VDP output can be accounted for by variations in noise between the two images, including edge effects during the wavelet transform process; however, the location and frequency of these deviations is not sufficient to cause a ruling of dissimilarity by human subjective tests.

The A-HNN predictive output was evaluated by three methods that are based on human-visual-system concepts, and the network outputs were found to be similar to the target values expected. The A-HNN has the ability to predict one sensor from another,

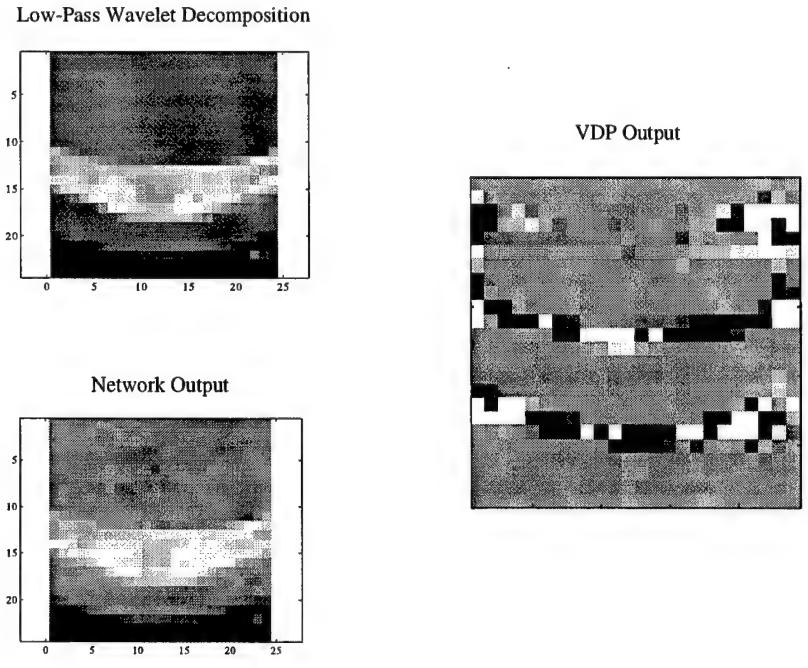



**Figure 31. VDP of A-HNN; CT data.**

and the autoassociative portion provides the capability to validate the overall network performance.

### *4.4 Conclusions*

This chapter presented experimental results, with features from the wavelet decomposition of the images being used as input to the A-HNN. Wavelet decomposition allowed determination of distinct, specific features from each of the CT and MR images, namely, bone and soft tissue. The A-HNN allowed prediction of one image from another. Additionally, the autoassociative portion of the A-HNN provided a means of validating the operation of the network by examining its stability. Utilizing the A-HNN provides a network with improved training performance over conventional multi-layer perceptrons. This is true for not only image processing, but also classification problems. Two additional applications for the A-HNN are presented in Appendix D -- a simple classification problem using XOR data, and a real-world materials analysis problem.

# 5 *Conclusions*

## *5.1 Research Contributions*

The goal of this research was to develop a computational method of mimicking the way in which a human analyzes two different image representations of the same material. The effort resulted in a computational means of extracting specific features from each of two images and a neural network for predicting one image from another, while also providing an autoassociative output with which to judge the stability and resolution performance of that prediction. Concepts of the human visual system were implemented in determining the features, and a new architecture, the Autoassociative-Heteroassociative Neural Network, was developed to associate the different image data sets.

Based on the transformation properties of the human visual system, Fourier, Gabor and wavelet transforms were examined for feature extraction. In this case the image data were CT and MR images of a head, and the two features to be extracted were bone and soft tissue. These features were extracted using the Daubechies $W_4$ wavelet transform.

Determining the association of the features was the next step. In keeping with the inspiration of the human machine, the objective was to process the human-visual-system-determined features by a neural network. The AANN was examined because of its use for image processing; however, AANNs have several limitations: 1) only one image at a time is processed, 2) the data processed are real-valued data, and 3) training by the global measure of mean squared error provides a nonlinear principal-components-analysis

equivalent. For this research, AANNs were extended beyond their general applications in several areas. A new interpretation, AANNs being used as filters, was developed. Since prior applications of AANNs involved real, single-valued data only, options for using AANNs to process complex-valued data were developed and demonstrated. This effort also proposed an alternative error function, the Visual-Difference Predictor, for training AANNs, which is based on human-visual-system concepts and is designed specifically for processing image data. This error function offers the option of training the network to similar or "close enough to be the same" by human visual standards.

This effort further extended the applications of AANNs from processing one image data set to processing two. A new architecture, the Autoassociative-Heteroassociative Neural Network, was developed to provide the ability to predict one sensor image from another. This new architecture provides both the autoassociative and predictor outputs, permitting validation of overall network operation as well as improvement of the training performance.

Research results were described on two fronts -- theoretical and practical. A theoretical investigation was conducted on the use of AANNs as filters and a new architecture was developed for use in predicting one sensor image from another. Three practical investigations were conducted. The first included examination of concepts from the human visual system for determining means of extracting features from image data. It was found that wavelets could be used for extracting features from the CT and MR images under investigation. Additionally, while investigating the Fourier-transformation space, options for processing complex-valued data through AANNs were developed, and an alternative error function for training image-processing AANNs was proposed.

## *5.2 Future Work*

Since neural networks are effective tools for processing information from different types of image sources, future work will be focused on using the design principles of neural systems to extract features and to process image data generated by different material process-control sensors. In the Air Force Research Laboratory, the need exists for controlling the manufacturing processes in the development of novel advanced materials. The data acquired from different material-processing techniques are generated from several different types of sensors. The information/features from each sensor must be extracted and missing information identified; the sensor information must be correlated for development of the feedback necessary to control the material manufacturing process.

The research documented here can be extended to materials process control where feature-extraction techniques are needed for isolating information from multiple sensors. Data-reduction techniques are required for visualizing the multi-dimensional sensor data currently being obtained. Methods of reducing the data by feature extraction or by artificial neural networks must be examined for each specific process of interest to the Air Force. Once the data are thoroughly analyzed, the feedback necessary to control the processes can be implemented.

*Appendix A - Visual-Difference Predictor as an Alternative Error Function for the AANN*

The most common error function for backpropagation training of an AANN is the global measure of network energy called the mean squared error (MSE). Backpropagation of the MSE error function is a means of driving the network output values to the smallest mean-squared deviations of the data values. The backpropagation algorithm employs gradient descent that follows the slope of the error surface; this surface can be highly convoluted, especially for variable data such as images that require a large number of training epochs for the convergence of the weights (Cottrell and others, 1989; Hecht-Nielsen, 1990). This appendix describes an alternative error function for training image-processing AANNs; this error function is called the Visual Difference Predictor (VDP).

The VDP is based on human-visual-system perception and is a means of evaluating images for similarity, or "close enough to the same image" by human perceptual standards. Human evaluation of the similarity between two images is an intricate process within the human visual system and is unrelated to specific individual pixel values or to the global calculation of MSE. On the other hand, MSE trains to a lossy version of the original image; because of the assumption of truncated eigenvectors, the VDP trains the network such that the weight updates continue until the pixels are similar, or close enough. The error function being proposed here assesses the difference between the network output and the target data, based on the principles of human evaluation of images.

The VDP model shown in Figure A.1 is suggested by the actual physiology of the human visual system, and the performance parameters are set by experimentally

determined psychophysical performance results. The VDP algorithm produces a map that predicts where visual differences will be perceived between two images at each pixel location. The result is the probability *P(c)* of detecting a signal of contrast *c*. The function that describes this probability is a Weibull probability distribution function and is given by (Nachmias, 1981):

$$P(c) = 1 - e^{-(c/\alpha)^{\beta}} \tag{A.1}$$

where $\alpha$ and $\beta$ are parameters based on a psychometric function -- contrast versus probability of detection. The plot of this psychometric function is a unified result of many experiments conducted in an attempt to describe the contrast detection for a specific spatial frequency (Daly, 1993). The curve resembles a sigmoid, having a domain of normalized contrast units from 0 to 2 and a range for probability of detection from 0 to 1; $\alpha$ controls the transition region, and $\beta$ controls the slope. $\beta$ is roughly a constant value, and $\alpha$ corresponds to a detection threshold between the original and the distorted image.

The "Cortical Stage" of the model (see Figure A.1) is implemented by a Gabor expansion. Each band-pass Gabor filter constitutes a cortex band in the model. Each cortex band (or subband) represents an orientation and a radial-frequency bandwidth, and the probability of detection for each pixel in the cortex band, at index *[I, J]*, is:

$$P_q[I,J] = 1 - \exp\left[-\left(\frac{\Delta C_q[I,J]}{T^q_{em}[I,J]\cdot T(0)}\right)^{\beta}\right] \tag{A.2}$$

where $q$ represents the subband index, and $\Delta C_q[I,J]$ is the difference in contrast between the input and the distorted image at pixel *[I,J]* of subband $q$. $T^q_{em}[I,J]$ is a mutual masking function (basically a thresholding value used when comparing the two images)

for determining the minimum threshold value between the images, and $T(0)$ is a threshold estimate of the internal noise of the visual detection mechanisms -- initially assumed to be 1.0 (Martin, 1996).

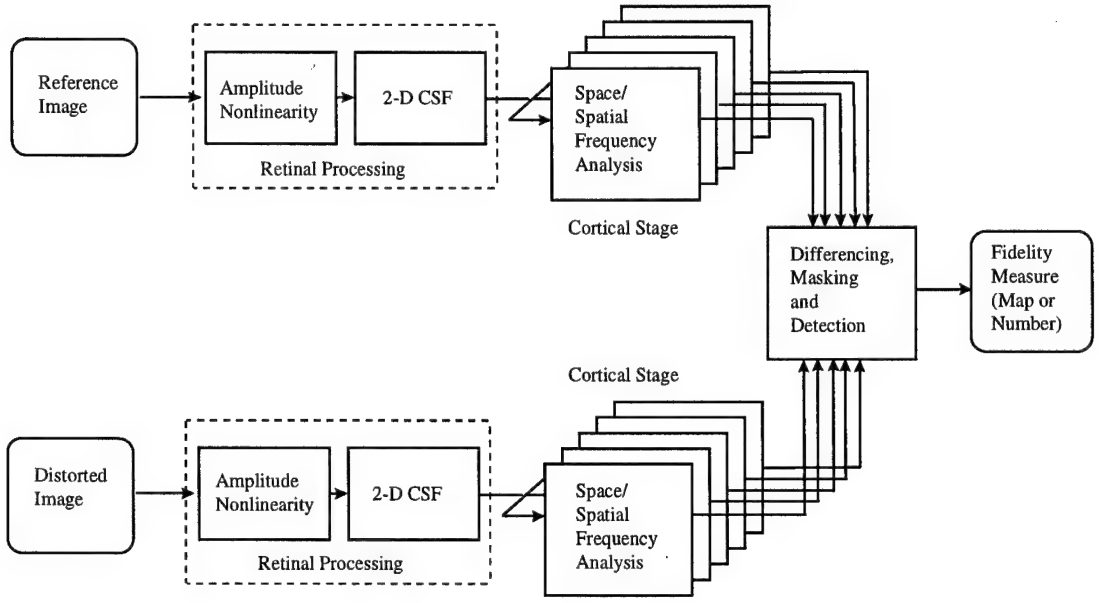



**Figure A.1. Common Structure of Image-Fidelity Measurements (VDP).**

The individual subband probabilities of detection are combined into a single image by

$$P_{Total}[I,J] = 1 - \prod_{q=1}^{Subbands} \left(1 - P_q[I,J]\right) \tag{A.3}$$

The fidelity measure is a quantitative metric for evaluating the similarity of two images -- an original and a distorted version -- at each pixel location. This fidelity map highlights locations where the associated pixels of the images are similar based on a human perceptual threshold, i.e., where a human would evaluate the pixels as close enough to be the same. Values near $\pm 1$ are considered to represent dissimilar pixels, and values near 0 are considered to represent similar. A sign is incorporated into the algorithm to identify the direction of the difference between the images.

As an error function for training neural networks, this fidelity map provides a pixel-by-pixel evaluation of the variation of the original or target image from the distorted or network output image. In regions where the fidelity map indicates too much variation from similar, the network will continue training. Using backpropagation, this metric seeks to drive the probability of detecting a difference in the images below the detection threshold level. This is feasible because the fidelity, or probability of detection ($P_D$), map is an isomorphism from the original image space to the probability space; therefore, the results of the $P_D$ can be used to identify those pixels that are "not similar enough", and update only those associated weights.

The AANN in Figure A.2 is shown here for purposes of identifying the notation used in the following weight update derivations for the VDP error function.

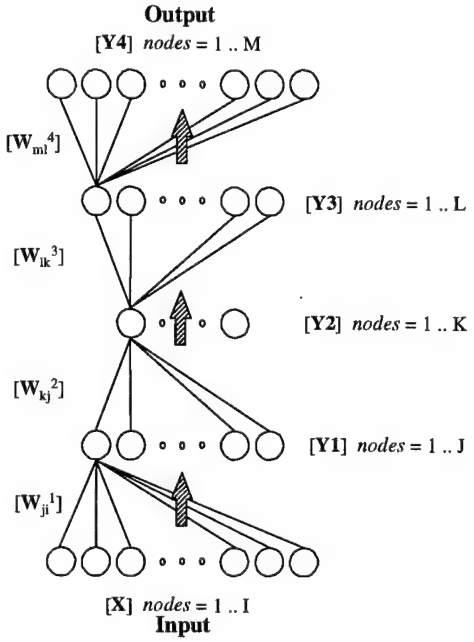



**Figure A.2. Three-Hidden-Layer AANN.**

The following is the generalized learning law for the backpropagation algorithm with the VDP metric:

$$w^{+} = w^{-} - \eta \frac{\partial P_{Total}(I,J)}{\partial w} \tag{A.4}$$

where

$$P_{Total}(I,J) = 1 - \prod_{q=1}^{Subbands} (1 - P_q[I,J]), \tag{A.5}$$

and each subband probability is as follows:

$$P_q[I,J] = 1 - \exp\left[ -\left( \frac{\Delta C_q[I,J]}{T_{em}^q[I,J] \cdot T(0)} \right)^{\beta} \right] \tag{A.6}$$

The contrast for each subband $q$ is

$$\Delta C_q[I,J] = d_q(I,J) - y4_q(I,J) \tag{A.7}$$

where $d$ is the input (or target output for an AANN), $y4$ is the network output, and $I,J$ is the pixel location for each subband. Each layer weight-update rule is generated with respect to the $\Delta C_q[I,J]$, since this is the only value having weights as a dependent variable.

The weight updates are derived as follows:

$$\frac{\partial}{\partial w} P_{Total}(I,J) = \frac{\partial}{\partial w} \left( 1 - \prod_{q=1}^{Subbands} \left( 1 - \left( 1 - \exp\left[ -\left( \frac{\Delta C_q[I,J]}{T_{em}^q[I,J] \cdot T(0)} \right)^{\beta} \right] \right) \right) \right) \tag{A.8}$$

With expansion, this becomes

$$\frac{\partial}{\partial w} P_{Total}(I,J) = \frac{\partial}{\partial w} \left( 1 - \prod_{q=1}^{Subbands} \left( 1 - \left( 1 - \exp\left[ -\left( \frac{\Delta C_q[I,J]}{T_{em}^q[I,J] \cdot T(0)} \right)^{\beta} \right] \right) \right) \right) \tag{A.9}$$

$$\frac{\partial}{\partial w}P_{Total}(I,J)=\frac{\partial}{\partial w}-\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=1}[I,J]}{T_{em}^{q=1}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=2}[I,J]}{T_{em}^{q=2}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\bullet\bullet \tag{A.10}$$

$$\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=SB}[I,J]}{T_{em}^{q=SB}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)$$

By the chain rule,

$$\frac{\partial}{\partial w}P_{Total}(I,J)=\frac{\partial}{\partial w}-\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=1}[I,J]}{T_{em}^{q=1}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=2}[I,J]}{T_{em}^{q=2}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\bullet\bullet \tag{A.11}$$

$$\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=SB}[I,J]}{T_{em}^{q=SB}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)$$

$$+\frac{\partial}{\partial w}-\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=2}[I,J]}{T_{em}^{q=2}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=1}[I,J]}{T_{em}^{q=1}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\bullet\bullet\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=SB}[I,J]}{T_{em}^{q=SB}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)+$$

$$\bullet\bullet\bullet+\frac{\partial}{\partial w}-\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=SB}[I,J]}{T_{em}^{q=SB}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=1}[I,J]}{T_{em}^{q=1}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\bullet\bullet\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=SB-1}[I,J]}{T_{em}^{q=SB-1}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)$$

Summarizing the notation,

$$\frac{\partial}{\partial w}P_{Total}(I,J)=\frac{\partial}{\partial w}-\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=1}[I,J]}{T_{em}^{q=1}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\prod_{q=2}^{SB}\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q}[I,J]}{T_{em}^{q}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right) \tag{A.12}$$

$$+\frac{\partial}{\partial w}-\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=2}[I,J]}{T_{em}^{q=2}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\prod_{\substack{q=1\\q\neq 2}}^{SB}\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q}[I,J]}{T_{em}^{q}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)+\bullet\bullet\bullet$$

$$+\frac{\partial}{\partial w}-\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q=SB}[I,J]}{T_{em}^{q=SB}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\prod_{\substack{q=1\\q\neq SB}}^{SB}\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q}[I,J]}{T_{em}^{q}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)$$

Collecting terms produces

$$\frac{\partial}{\partial w}P_{Total}(I,J)=\sum_{q=1}^{SB}\frac{\partial}{\partial w}-\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{q}[I,J]}{T_{em}^{q}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\prod_{\substack{r=1\\r\neq q}}^{SB}\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_{r}[I,J]}{T_{em}^{r}[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right) \tag{A.13}$$

Rearranging terms yields

$$\frac{\partial}{\partial w}P_{Total}(I,J)=\sum_{q=1}^{SB}(-1)\prod_{\substack{r=1\\ r\neq q}}^{SB}\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_r[I,J]}{T_{em}^r[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)\bullet\frac{\partial}{\partial w}\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right) \tag{A.14}$$

$$\frac{\partial}{\partial w}P_{Total}(I,J)=\sum_{q=1}^{SB}(-1)\prod_{\substack{r=1\\ r\neq q}}^{SB}(1-P_r[I,J])\bullet\frac{\partial}{\partial w}\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right) \tag{A.15}$$

The partial derivative still to be taken is

$$\frac{\partial}{\partial w}\left(1-\left(1-\exp\left[-\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta}\right]\right)\right)=\frac{\partial}{\partial w}\exp\left[-\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta}\right] \tag{A.16}$$

$$\frac{\partial}{\partial w}\exp\left[-\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta}\right]=\exp\left[-\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta}\right]\cdot\frac{\beta}{T_{em}^q[I,J]\cdot T(0)}\cdot\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta-1}\frac{\partial}{\partial w}\Delta C_q[I,J] \tag{A.17}$$

Collecting terms results in

$$\frac{\partial}{\partial w}P_{Total}(I,J)=\sum_{q=1}^{SB}(-1)\prod_{\substack{r=1\\ r\neq q}}^{SB}(1-P_r[I,J])\bullet\exp\left[-\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta}\right]\cdot\frac{\beta}{T_{em}^q[I,J]\cdot T(0)}\cdot\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta-1}\frac{\partial}{\partial w}\Delta C_q[I,J] \tag{A.18}$$

For compactness of representation, the collection of terms that are calculated for each pixel value, *[I,J]*, in each subband *q*, is

$$\mathrm{K}_{[I,J]}^{q}=\sum_{q=1}^{SB}(-1)\prod_{\substack{r=1\\ r\neq q}}^{SB}(1-P_r[I,J])\bullet\exp\left[-\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta}\right]\cdot\frac{\beta}{T_{em}^q[I,J]\cdot T(0)}\cdot\left(\frac{\Delta C_q[I,J]}{T_{em}^q[I,J]\cdot T(0)}\right)^{\beta-1} \tag{A.19}$$

Therefore the partial derivative will have the form

$$\frac{\partial}{\partial w}P_{Total}(I,J)=\mathrm{K}_{[I,J]}^{q}\cdot\frac{\partial}{\partial w}\Delta C_q[I,J] \tag{A.20}$$

The term containing the dependent variable, the weights, is

$$\Delta C_q[I,J]=d_q(I,J)-y4_q(I,J) \tag{A.21}$$

This value represents the error between the network output and the target values, and is very similar in format to the MSE metric typically used in backpropagation (Wasserman, 1989). From this point, the derivation of the weight equations is straightforward.

The output-layer weights (or fourth layer weights) are designated by subscripts $m$ and $l$ and superscript 4 and specify the connection between the $l^{th}$ node in the third hidden layer and the $m^{th}$ node in the output layer. Each output from the output layer is designated by $y4_k$. From the generalized form of the learning law, the updated weight is established as

$$w^{4+}_{m_0,l_0} = w^{4-}_{m_0,l_0} - \eta \frac{\partial P_{Total}}{\partial w^4_{m_0,l_0}} \tag{A.22}$$

To implement this equation, the partial derivative of the probability of detection, $P_{Total}$, with respect to the fourth-layer weights is evaluated as follows:

$$\frac{\partial P_{Total}}{\partial w^4_{m_0,l_0}} = \mathrm{K}^q_{[I,J]} \cdot \frac{\partial}{\partial w^4_{m_0,l_0}} \Delta C_{k,l}(I,J) \tag{A.23}$$

Therefore,

$$\frac{\partial}{\partial w^4_{m_0,l_0}} \Delta C_q(I,J) = \frac{\partial}{\partial w^4_{m_0,l_0}} \left(d_{m_0} - y4_{m_0}\right) = \frac{\partial\left(-y4_{m_0}\right)}{\partial w^4_{m_0,l_0}} \tag{A.24}$$

where $y4_{m0}$ can be written in the generalized form

$$\frac{\partial y4_{m_0}}{\partial w^4_{m_0,l_0}} = \frac{\partial}{\partial w^4_{m_0,l_0}} F\left(\sum_{l=1}^{L+1} w^4_{m_0,l} y3_l\right) = y3_{l_0} \tag{A.25}$$

with $F$ being the activation function on the output-layer nodes. Here, the activation function is the linear function, and the only partial derivative that survives is the term associated with $m_0$ and $l_0$. Thus, the output (fourth)-layer-weight update formula is

$$w_{m_0,l_0}^{4+} = w_{m_0,l_0}^{4-} + \eta \cdot \mathrm{K}_{[I,J]}^{q} \cdot y3_{l_0} \tag{A.26}$$

To obtain the weights into the third hidden layer, the same derivation process must be performed. The third-layer weights are designated by subscripts $l$ and $k$ and superscript 3 and specify the connection between the $k^{\text{th}}$ node in the second hidden layer and the $l^{\text{th}}$ node in the third hidden layer. For the third hidden layer, the activation function is a sigmoid

$$F(x) = \frac{1}{1+e^{-x}} \tag{A.27}$$

and taking the derivative of a sigmoid yields

$$F'(x) = (-1)(1+e^{-x})^{-2}(e^{-x}) = -\frac{e^{-x}}{(1+e^{-x})^2} = F(x)(1-F(x)) \tag{A.28}$$

Therefore,

$$\frac{\partial P_{Total}}{\partial w_{l_0,k_0}^{3}} = -\sum_{m=1}^{M} \mathrm{K}_{[I,J]}^{q} \cdot w_{m,l_0}^{4} \cdot y3_{l_0}(1-y3_{l_0}) \cdot y2_{k_0} \tag{A.29}$$

By substitution, the third-layer weights are updated by

$$w_{l_0,k_0}^{3+} = w_{l_0,k_0}^{3-} + \eta \sum_{m=1}^{M} \mathrm{K}_{[I,J]}^{q} \cdot w_{m,l_0}^{4} \cdot y3_{l_0}(1-y3_{l_0}) \cdot y2_{k_0} \tag{A.30}$$

The second-layer weights are designated by subscripts $k$ and $j$ and superscript 2 and specify the connection between the $j^{\text{th}}$ node in the first hidden layer and the $k^{\text{th}}$ node in the second hidden layer. The activation function on the second hidden layer, the bottleneck node, is linear thus, the second-layer weights update as

$$w_{k_0,j_0}^{2+} = w_{k_0,j_0}^{2-} + \eta \sum_{m=1}^{M}\sum_{l=1}^{L} \mathrm{K}_{[I,J]}^{q} \cdot w_{m,l}^{4} \cdot y3_{l}(1-y3_{l}) \cdot w_{l,k_0}^{3} \cdot y1_{j_0} \tag{A.31}$$

The first layer weights are designated by subscripts $j$ and $i$ and superscript 1 and specify the connection between the $i^{th}$ node on the input and the $j^{th}$ node in the first hidden layer. The activation function on the first hidden layer is a sigmoid, and the first layer weights update as

$$w_{l_0,k_0}^{1+} = w_{l_0,k_0}^{1-} + \eta \sum_{m=1}^{M} \sum_{l=1}^{L} \sum_{k=1}^{K} \mathrm{K}_{[I,J]}^{q} \cdot w_{m,l}^{4} \cdot y3_l (1 - y3_l) \cdot w_{l,k}^{3} \cdot w_{k,j_0}^{2} \cdot y1_{j_0} (1 - y1_{j_0}) \cdot x_{i_0} \tag{A.32}$$

These weight-update rules are designed specifically for training networks using image data. Here the error function is related to the concept of training to similarity rather than to absolute pixel values. The VDP can be a very effective metric for training a neural network to process images. The results are specifically optimized for the human visual system.

*Appendix B - Neural-Network Processing of Complex-Valued Data*

## *B.1 Fourier Coefficients and Complex AANNs*

The Fourier transform of a gray-scale real-valued image results in a matrix of complex-valued numbers that represent the magnitude and phase of the frequency components of the image. Neural networks were designed to process real, single-valued numbers. To deal with complex-valued data such as Fourier coefficients, additional processing must be considered. When Fourier coefficients are used as input, a "complex" neural network is required that can maintain the complex relationship that exists in the frequency domain. This appendix describes three methods of implementing a complex-valued neural network.

## *B.2 Complex Inputs*

The first implementation involves an AANN with complex values on the input, the weight matrices randomly initialized as scalar values, and the weight updates maintained as scalar updates. Figure B.1 graphically depicts data processing via the AANN. The pixel values of the original image are transformed to Fourier space. The Fourier coefficients (complex-valued numbers) are applied as input to the AANN.

When scalar weight matrices are used, the real and complex components are treated identically through the network, ensuring that the magnitude/phase relationship of the input data is maintained. It is desirable to maintain the ordered-pair relationship that

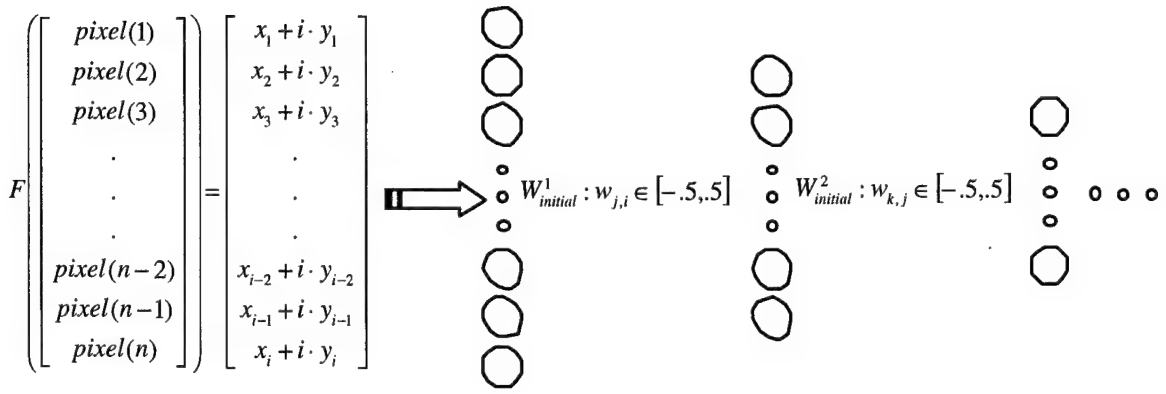



**Figure B.1. Complex-Valued Input and Scalar-Weight Matrices.**

exists between the real and imaginary components of the data. Therefore, the weights must be scalar, to ensure that changes will not occur in only one element of the ordered pair.

A three-hidden-layer AANN was generated, with sigmoid activation functions on the first and third hidden layers and linear activation functions on the middle hidden layer and the output. The input and output data were complex-valued numbers of the form $x+i\cdot y$. Figure B.2 is a schematic of this AANN architecture, showing the various parameters of the network.

A systematic method of training multilayer ANNs is backpropagation. The backpropagation algorithm, which is a deterministic training method, employs a type of gradient descent; that is, it follows the slope of the error surface downward, constantly adjusting the weights toward a minimum. Backpropagation requires that the partial derivative of the error, *E*, be computed with respect to each weight. The most common

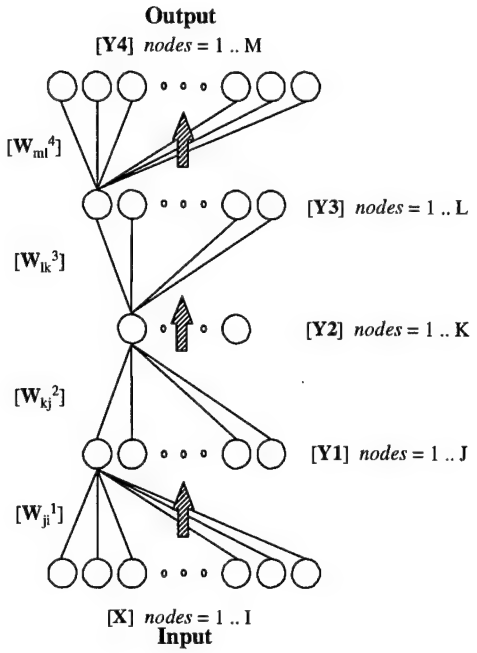



**Figure B.2. Three-Hidden-Layer AANN.**

error function used for the global measure of network energy is the mean squared error (MSE) which is defined as

$$E \cong \frac{1}{2}\sum_{m=1}^{M}(d_m - y4_m)^2 \tag{B.1}$$

where $d_m$ is the target output and $y4_m$ is the network output.

Each output for the network can be defined as a function of a set of network parameters, namely, the network weights. The generalized learning law for each set of weights is

$$[W^+] = [W^-] - \eta \frac{\partial E}{\partial [W^-]} \tag{B.2}$$

where $[W^+]$ is the updated weight set, $[W^-]$ is the old weight set, and $\eta$ is the learning rate for the backpropagation.

The output layer weights (or fourth-layer weights) are designated by subscripts $m$ and $l$ and superscript 4 and specify the connection between the $l^{th}$ node in the third hidden layer and the $m^{th}$ node in the output layer. Each output from the output layer is designated by $y4_k$. From the generalized form of the learning law, the updated weight is established as

$$w^{4+}_{m_0,l_0} = w^{4-}_{m_0,l_0} - \eta \frac{\partial E}{\partial w^4_{m_0,l_0}} \tag{B.3}$$

To implement this equation, the partial derivative of the error $E$, with respect to the weights, is evaluated using

$$\frac{\partial E}{\partial w^4_{m_0,l_0}} = \frac{\partial}{\partial w^4_{m_0,l_0}} \left\{ \frac{1}{2} \sum_{m=1}^{M} (d_m - y4_m)^2 \right\} \tag{B.4}$$

In the summation, the dependence on $w_{m_0 k_0}$ must be isolated. Expanding the summation yields

$$\frac{1}{2} \sum_{m=1}^{M} (d_m - y4_m)^2 = \frac{1}{2} \left\{ (d_1 - y4_1)^2 + \ldots + (d_{m_0} - y4_{m_0})^2 + \ldots + (d_m - y4_m)^2 \right\} \tag{B.5}$$

The only partial derivatives to survive the differentiation are variables that involve both subscripts $m_0$ and $k_0$. While the desired output $d_m$ is a constant, the network output $y4_m$ is a function of the weighted outputs from the hidden layer. The partial derivative of the summation simplifies to

$$\frac{\partial (d_m - y4_m)^2}{\partial w^4_{m_0,l_0}} = \left\{ \begin{array}{l} 0 : m \neq m_0 \\ 2(d_{m_0} - y4_{m_0}) \dfrac{\partial (-y4_{m_0})}{\partial w^4_{m_0,l_0}} : m = m_0 \end{array} \right\} \tag{B.6}$$

and, therefore,

$$\frac{\partial E}{\partial w^4_{m_0,l_0}} = \frac{\partial}{\partial w^4_{m_0,l_0}}\left\{\frac{1}{2}\sum_{m=1}^{M}(d_m - y4_m)^2\right\} = (d_{m_0} - y4_{m_0})\frac{\partial(-y4_{m_0})}{\partial w^4_{m_0,l_0}} \tag{B.7}$$

where $y4_{m0}$ can be written in the generalized form

$$\frac{\partial y4_{m_0}}{\partial w^4_{m_0,l_0}} = \frac{\partial}{\partial w^4_{m_0,l_0}} F\left(\sum_{l=1}^{L+1} w^4_{m_0,l}\, y3_l\right) = y3_{l_0} \tag{B.8}$$

with $F$ being the activation function on the output-layer nodes. Here, the activation function is the linear function. Again, the only partial derivative to survive is the term associated with $m_0$ and $l_0$. By substitution

$$\frac{\partial E}{\partial w^4_{m_0,l_0}} = -(d_{m_0} - y4_{m_0})y3_{l_0} \tag{B.9}$$

Thus, the output (fourth)-layer weight-update formula is

$$w^{4+}_{m_0,l_0} = w^{4-}_{m_0,l_0} + \eta(d_{m_0} - y4_{m_0})y3_{l_0} \tag{B.10}$$

For all remaining weight updates, the same derivation process is performed. For the third hidden layer, the weights are designated by subscripts $l$ and $k$ and superscript 3 and specify the connection between the $k^{\text{th}}$ node in the second hidden layer and the $l^{\text{th}}$ node in the third hidden layer. For the third hidden layer, a sigmoid was considered for use as the activation function since it has been shown to have the same universal approximation power as the Fourier series (Cybenko, 1989). However, the proof has been demonstrated only when the sigmoid is a real single-valued function and is bounded on the output. These criteria make the sigmoid function dense in the space of continuous functions -- a property necessary for universal approximation (Cybenko, 1989). In the present study, the data are neither real nor single valued but complex, with an output of an ordered pair of complex-valued numbers. The sigmoid function with complex inputs has the form

$$\frac{1}{1+e^{-z}} = \frac{1}{1+e^{-x-i\cdot y}} = \frac{1}{1+e^{-x}e^{-i\cdot y}} \tag{B.11}$$

The function is bounded if the denominator does not equal zero. With real-valued inputs, this is the case; with complex inputs, however, discontinuities may exist in the complex plane. When $y = \pm(2n-1)\pi$, then $e^{-i\cdot y} = (-1)$. When $x=0$, the denominator is zero. Therefore, discontinuities in the complex plane occur cyclically at all odd-numbered $\pi$ values when $x=0$. However, within the domain $y \in (-\pi, \pi)$, the complex sigmoid is bounded on the output complex plane. Since the output is bounded, at least within a certain input domain, then the sigmoid can still be used as a universal approximator (Oxley and Suter). Before being processed through the network, each of the Fourier-coefficient training vectors is energy-normalized, to lie within the unit circle. With this normalization, $\|(x, y)\| \leq 1$, the input is bounded and less than $\pi$, which means that the function is bounded on its output. Within the present application domain, the sigmoid can be employed as an activation function and was selected for use on the third hidden layer. The derivative of a sigmoid is calculated as

$$F'(\alpha) = (-1)(1+e^{-\alpha})^{-2}(e^{-\alpha}) = -\frac{e^{-\alpha}}{(1+e^{-\alpha})^2} = F(\alpha)(1-F(\alpha)) \tag{B.12}$$

By substitution, the third-layer weights are updated by

$$w^{3+}_{l_0,k_0} = w^{3-}_{l_0,k_0} + \eta \sum_{m=1}^{M} (d_m - y4_m)\cdot w^4_{m,l_0} \cdot y3_{l_0}(1-y3_{l_0})\cdot y2_{k_0} \tag{B.13}$$

The second-layer weights are designated by subscripts $k$ and $j$ and superscript 2 and specify the connection between the $j^{th}$ node in the first hidden layer and the $k^{th}$ node in the second hidden layer. The activation function on the second hidden layer (the bottleneck node) is linear, and the second-layer weights are updated by

$$w_{k_0,j_0}^{2+} = w_{k_0,j_0}^{2-} + \eta \sum_{m=1}^{M} \sum_{l=1}^{L} (d_m - y4_m) \cdot w_{m,l}^{4} \cdot y3_l (1 - y3_l) \cdot w_{l,k_0}^{3} \cdot y1_{j_0} \tag{B.14}$$

The first layer weights are designated by subscripts $j$ and $i$ and superscript 1 and specify the connection between the $i^{th}$ node on the input and the $j^{th}$ node in the first hidden layer. The activation function on the first hidden layer is a sigmoid; thus, the partial derivative of the function is as stated above, and the first-layer weights are updated by

$$w_{l_0,k_0}^{1+} = w_{l_0,k_0}^{1-} + \eta \sum_{m=1}^{M} \sum_{l=1}^{L} \sum_{k=1}^{K} (d_m - y4_m) \cdot w_{m,l}^{4} \cdot y3_l (1 - y3_l) \cdot w_{l,k}^{3} \cdot w_{k,j_0}^{2} \cdot y1_{j_0} (1 - y1_{j_0}) \cdot x_{i_0} \tag{B.15}$$

Since *y1* - *y4* are complex-valued numbers, the weight updates will also be complex-valued numbers. However, the weights must remain scalar values to ensure that the network is maintaining the ordered-pair relationship of the original input data. Therefore, the update values for [**W**] must be made real and scalar. The scalar value can be obtained from either the real or the imaginary value for the weight update or from the magnitude of the complex-valued weight update. When the real or imaginary terms are used as the scalar-weight update, only one-half of the update information is being utilized, and either the real or the phase information is being ignored. The magnitude of the complex-valued weight update incorporates the influence of both the real and the imaginary information, although the phase information is lost. However, when the magnitude is calculated, the result is real and positive. The direction of the descent to the minimum value on the error surface is masked by this calculation and must be determined from the signs of both the real and the imaginary terms. However, since the weight update is real and scalar, a "combined" gradient-descent direction must be determined from the complex-valued weight updates. The combined direction will have either a

positive or a negative sign and that sign would be correct only for those complex-valued data in the first (positive) and third (negative) quadrants. For the second and fourth quadrants, where the complex-valued data have mixed signs, the sign selected would not be representative of both elements of the ordered pair.

The appropriate sign, e.g., real-term sign, imaginary-term sign, or a combination, was experimentally determined to be the sign of the real component. The network converges to a training error of < 1.0 in 20 training epochs when the sign of the real term is used. When the sign of the imaginary term is used, an extremely small learning rate and a very slow descent results; after 500 epochs the training error is still very high, and the network has not learned to generate the output from the input autoassociatively. The information from the initial weight-update calculation is still only partially incorporated. When the sign of the real component only is applied, the sign of the imaginary term may not be correct. A change in sign on the imaginary portion of the weight update means that the complex-valued term in the original data is 180 degrees out of phase.

Figure B.3 is a representative image from the training population for developing an AANN. Note that the image is basically a three-gray-scale image of a cross, with the image varying from white on the outside to black on the inside.

Thirty-one input images similar to the one in Figure B.3, each having a different resolution and added noise, were pre-processed into the Fourier space. The Fourier transform, being a unitary transform, provides an equivalence to the pixel data. The three-hidden-layer AANN was then trained on a subset of the Fourier coefficients. It was determined empirically that a kernel of 23 coefficients surrounding the DC value of the image would be sufficient to replicate the image when the inverse Fourier transform was

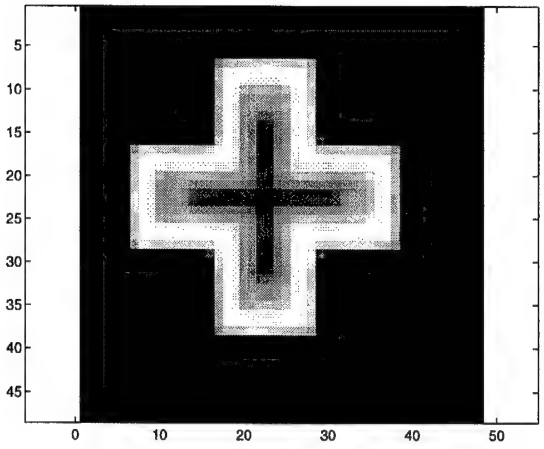



**Figure B.3. Three-Gray-Level Cross-Image Sample Input.**

applied. The input to the AANN is the 23 x 23 matrix of coefficients with the DC value at the center. This 23 x 23 matrix produces 529 complex values. However, the symmetry of the Fourier transform is as follows; the upper left quadrant of the matrix is the complex conjugate of the lower right quadrant, and the upper right quadrant is the complex conjugate of the lower left. In this known symmetry, only one-half of the Fourier coefficients are unique and must be processed. These 265 complex numbers were vectorized and used to train the AANN, with 132.5:1 compression on the bottleneck (two nodes). The network was trained with 31 training samples, each being an image of a three-gray-level cross that had been Fourier processed and vectorized. Through trial and error, the AANN was implemented as follows: 265 complex numbers on the input, 15 nodes on the first hidden layer, two nodes on the bottleneck, 15 nodes on the third hidden layer, and 265 complex numbers on the output. The activation functions chosen for the first and third hidden layers were sigmoids, and a linear activation function was selected for the bottleneck layer -- typical choices for an AANN in a data-compression application (Kramer, 1991). With a fully trained network, this architecture results in an output with

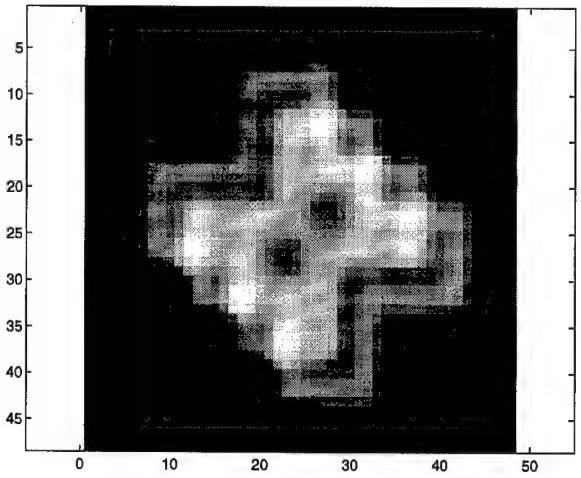



**Figure B.4. Output of Neural Network Trained with Complex Inputs.**

two overlapping images -- the original image and an image 180 degrees out of phase. Figure B.4 is an example of the output of the trained network.

*Conclusion* This figure displays an unsatisfactory result for an AANN which has as its purpose to recreate the original data set and provide a compressed representation of the data on the bottleneck layer. Use of the scalar-weight matrices with the complex-valued data prevents application of the appropriate gradient-descent direction to both members of the ordered pair. As a result, an echo image appears that is 180 degrees out of phase. One method of avoiding the out-of-phase image would be to allow the weight matrices to remain complex. However, this would alter the ordered-pair relationship between the real and imaginary terms in the original input data. Therefore, to process complex values through an AANN, the data must be formatted in such a way that all values input to the network are real and single valued.

### *B.3 Stacked Real/Imaginary Vector Input with Random Weight Matrices*

The second implementation of a complex-valued neural network involves an AANN with training vectors of real values on the input and the weight matrices randomly

initialized with scalar values. The training data are created by processing the image data via Fourier transforms, truncating to an appropriate kernel, and processing only the unique Fourier coefficients, as described above. Once the 265 unique complex-valued numbers have been identified, the training vector is created by stacking the real components of the 265 complex-valued numbers (top half of the input training vector) and the imaginary components (bottom half). The input training vectors are now 529 data values in length (since the DC term has a zero-valued imaginary component; there are 265 real components and 264 imaginary components). The main difference between this and the previous implementation is that the input training vectors in this case are real values. The weight matrices are initialized as random scalar matrices. Figure B.5 graphically depicts data processing through the AANN for this implementation.

A three-hidden-layer AANN was generated as described in the previous section, with 15 nodes on the first hidden layer, two nodes on the bottleneck, and 15 nodes on the

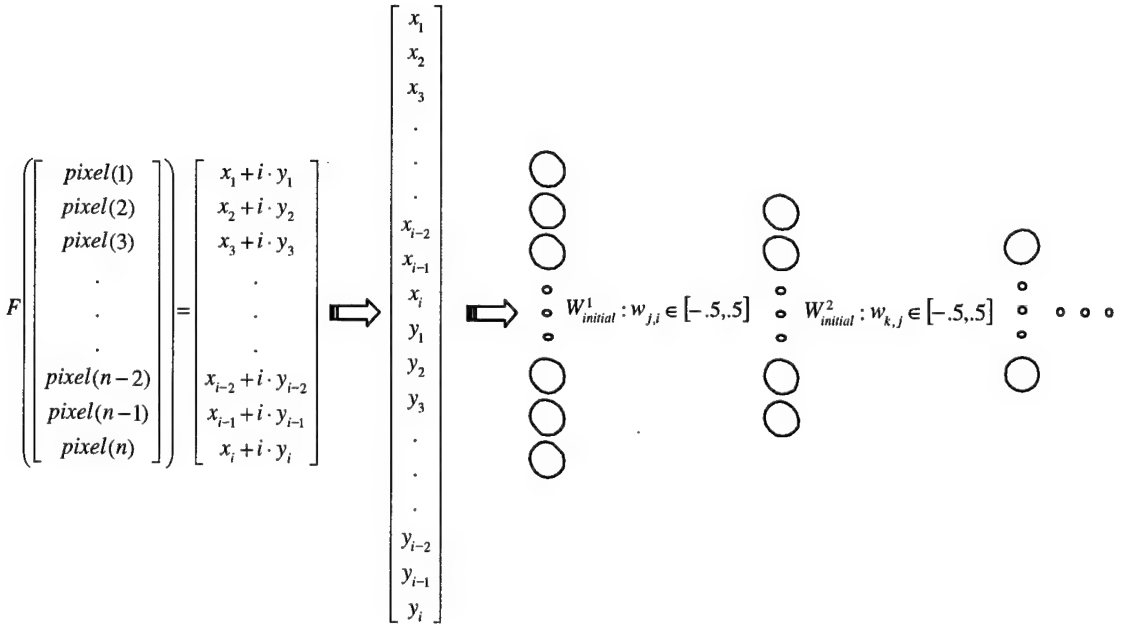



**Figure B.5. Stacked Real/Imaginary Inputs and Random-Weight Matrices.**

third hidden layer. Sigmoid activation functions were chosen for the first and third hidden layers and linear activation functions for the bottleneck hidden layer. Training was by backpropagation of the MSE function. This AANN converged in only two training epochs. Figure B.6 represents the output of this AANN trained on 31 images, with the data formatted as stacked real and imaginary terms with randomly-initialized scalar weights.

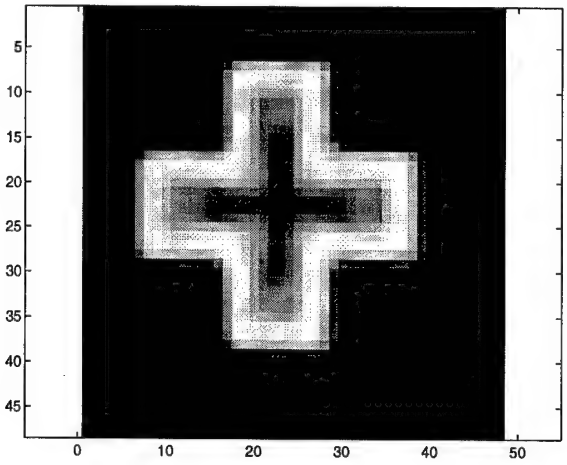



**Figure B.6. Output of Neural Network Trained with Stacked Real/Imaginary Inputs and Random Weights.**

*Conclusion* This method of stacking the real and imaginary coefficients into one contiguous input vector effectively and satisfactorily trains the AANN, with the reduced representation being on the bottleneck, using complex-valued data. Processing of the data to generate real, single-valued input vectors permits the use of all artificial neural networks and reduces the problem to one of conventional neural-network processing. This method of processing the complex data simplifies network implementation; however, some *a priori* information concerning the data has not been utilized -- the ordered-pair complex relationship of the input data. The next section will

examine initializing the network such that a prior knowledge of the complex relationship in the input data is incorporated.

### *B.4 Stacked Real/Imaginary Vector Input with Formatted Weight Matrices*

The third implementation of a complex-valued neural network involves an AANN with training vectors of real values as the input being obtained in the stacking manner described previously. In this case, the weight matrices are not initialized as random scalars but in a way that will ensure complex multiplication.

Given two complex numbers, $(x+i\cdot y)$ and $(u+i\cdot v)$,

$$(x+i\cdot y)\cdot(u+i\cdot v)=(x\cdot u-y\cdot v)+i\cdot(x\cdot v+y\cdot u)\rightarrow\begin{bmatrix}u & -v\\ v & u\end{bmatrix}\cdot\begin{bmatrix}x\\ y\end{bmatrix} \tag{B.17}$$

With the real components of the Fourier coefficients designated as $x$ and the imaginary components designated as $y$, the complex multiplication is performed if the weight matrices are defined in the form: $\begin{bmatrix}u & -v\\ v & u\end{bmatrix}$, where $u$ and $v$ are random scalars $\in [-.5,.5]$. This process will maintain the complex relationship by multiplication, without requiring the presence of a specific imaginary term. Here, the network trains on real numbers, but the imaginary component is maintained by forcing the multiplication rule. The network architecture described previously is used: 15 nodes on the first hidden layer, two nodes on the bottleneck, 15 nodes on the third hidden layer. Sigmoid activation functions are chosen for the first and third hidden layers and linear activation functions for the bottleneck hidden layer. This architecture results in the image in Figure B.7. This version also converges in only two training epochs.

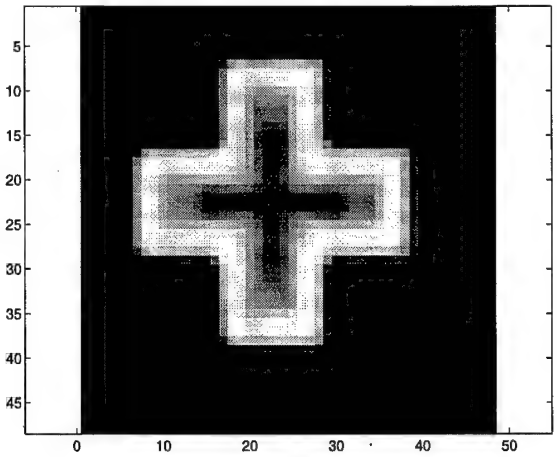



**Figure B.7. Output of Neural Network Trained with Stacked Real/Imaginary Inputs and Formatted Weights.**

*Conclusions* This method also effectively and satisfactorily trains the AANN, with the reduced representation being on the bottleneck, using complex-valued data. Again, the real single-valued input vectors permit the use of all artificial neural networks and reduce the problem to conventional neural-network processing.

This method of providing a specific initial format for the weights converges in the same rapid manner as the previous method but does not reduce the training time. Randomly initialized weight matrices, when fully trained, will converge to provide complex multiplication. Since the network converges, the ordered-pair relationship is maintained. There appears to be no advantage in pre-formatting the weights to enforce the complex multiplication.

## *B.5 Conclusions*

In the case of complex-valued data, neural networks can be very effective. Within specific bounds, the sigmoid activation function maintains its universal approximation power -- even when complex-valued data are applied. Thus, the proven concepts of artificial neural networks will apply not only to real data but also to complex-

valued data. For processing complex values through an AANN, the data must be formatted in such a way that input values to the network are real and single valued. An AANN cannot satisfactorily process complex-valued data, primarily because the direction of the gradient descent cannot be fully incorporated into the weight updates. Processing complex-valued data by stacking the real and imaginary coefficients into one contiguous input vector generates real, single-valued inputs to the AANN. This is an effective and satisfactory means of processing complex-valued data through an AANN, providing a reduced representation on the bottleneck. The weight matrices can be either initialized as random scalars or formatted to incorporate the ordered-pair complex multiplication relationship. As training progresses, both initializations converge rapidly, therefore, the set of initial weights used is of no consequence. Varying the format of the initialization of the weight matrices does not reduce the training time. The initialization formats will result in equivalent networks. Randomly initialized weight matrices, when fully trained, will converge to provide complex multiplication. Since the network converges, the ordered-pair relationship is obviously being maintained. The convergence of this data through the network is very rapid; thus, there appears to be no advantage in pre-formatting the weights to enforce the complex multiplication.

## *Appendix C - Trial and Error Process for Feature and Architecture Selection*

The trial and error process necessitates a trade-off among many factors. Feature selection, network architecture -- number of layers and number of nodes per layer -- and training time are the key elements in developing a neural network. The overall process is an iterative one, where multiple neural networks are programmed. Each programmed network alters one factor at a time, i.e., a sequence of network runs to examine different number of hidden layers or different number of nodes on each hidden layer. The result of each network run is evaluated against previous runs and the "best" performance is selected as the feature and architecture for the application. Determination of best is based on convergence of the network weights to the desired output, e.g., in a classification problem a confusion matrix would be examined; in an Autoassociative Neural Network (AANN) the mean-squared error (MSE) may be examined.

### *C.1 Feature and Architecture Selection Trial and Error Process Example*

For the gray-scale detection application (Section 3.3), the first consideration is determining the appropriate feature set. For compression processing of gray-scale images using AANNs, subimage tiling is commonly employed. The optimum size of the tiles for a specific application must be determined. For many image-processing applications, an 8 x 8 tile size is selected (Malthouse, 1996). The tile size is based on the size of the image data to be compressed and, or, by the resolution of the image and the detail to be maintained. The accepted tile size of 8 x 8 pixels becomes the starting point for tile size examination.

The Lenna image contains 48 x 48 pixels, and initial visual examination of the 8 x 8 tiles indicates that too much detail will be lost should the AANN be trained on such large areas of the original image. Smaller image tiles examined include 4 x 4 and 2 x 2, which are not eliminated upon visual study.

With potential features identified, the next step is to program multiple neural networks to examine the performance of each run as features, the number of hidden layers, and number of nodes per hidden layer are changed. For AANNs in compression applications, three-hidden layers are standard. The bottleneck layer, for compression, must be less than the number of features, and can be stipulated if a specific compression ratio is desired. If compression is not relevant, the bottleneck layer can have any number of nodes, but never needs more nodes than the number of features. The multiple network runs are now performed altering the number of nodes present on the first and third hidden layers and examining the choice of tile size for features.

For each network run, the convergence of the network weights is monitored by the MSE for each architecture configuration and feature. The convergence of weights requires a trade-off between network size and training time to converge (number of epochs). For this application, gray-scale detection on gray-scale images, the "best" feature and architecture selection was for tile size of 2 x 2, for an AANN with 15 nodes on the first and third layers and four nodes on the bottleneck layer.

### *C.2 Conclusions*

The selection process for features and architectures is primarily accomplished by trial and error. Incorporated into the trial and error process is utilization of as much *a*

*priori* information as is available about the specific problem or application. Oftentimes, knowledge of the problem provides a starting point for feature and architecture selection.

*Appendix D – Autoassociative-Heteroassociative Neural Network as a Classifier*

## *D.1 XOR Classification Data*

The fundamental concept of the A-HNN is to train the network to a desired output, while using inputs as target data. The discussion thus far has been surrounding the concepts of autoassociative and heteroassociative training, using image data. The addition of the input data to the target vector, not only provides a means of determining the generalization robustness of the network, but also improves the training performance of the network. This concept extends to work with classification data.

Multi-layer perceptrons, as described in Section 2.3.2, are used for classifying data using a class-labeled set of inputs. The fundamental concept of the A-HNN -- using input data as desired outputs -- will improve the training performance of the multi-layer perceptron for classification.

A 600-point set of class-labeled XOR data was generated for training a multi-layer perceptron and an A-HNN, 300 for training and 300 for testing. A single-hidden-layer network is sufficient to separate the two-class XOR data, therefore, the architecture for both types of multi-layer perceptrons will have two input nodes, a single hidden layer, and output. For classification problems, the criterion for best performance of the network is testing accuracy.

For a conventional multi-layer perceptron, after multiple program runs, the best performance architecture was chosen as a network with 10 nodes on the hidden layer and sigmoid activation functions. This resulted in a network with 100% testing accuracy, as shown in Table D.1 confusion matrix, trained in 1000 epochs to an MSE of 0.320767,

and required 78.6 million flops. The column headings of the table refer to the multi-layer perceptron output values, with thresholds applied at the class label values of $0 \pm 0.25$ and $1 \pm 0.25$; the undecided column would contain any network output values more than 0.25 away from 0 or 1, i.e., $0.25 <$ output value $< 0.75$. The row headings are the test data labels.

| Network Output → <br> Test Data Label ↓ | Class 0 | Class 1 | Undecided |
|---|---|---|---|
| Class 0 | 150 | 0 | 0 |
| Class 1 | 0 | 150 | 0 |

**Table D.1. Multi-Layer Perceptron Confusion Matrix for XOR Data.**

Next, multiple single-hidden-layer A-HNNs were generated. Different target output configurations as well as different number of nodes on the hidden layer were tested. An XOR data classifier has two inputs and one output. The program runs consisted of testing different combinations of the two-input values as target-output values. After multiple program runs, the best performance architecture was chosen as a network using the first input value as a target output value and five nodes on the hidden layer each with sigmoid activation functions. This resulted in a network with 100% testing accuracy, as shown in Table D.2 confusion matrix, trained in 1000 epochs to an MSE of 1.29785, and required 52.1 million flops. The same threshold values as described above were used.

| Network Output → Test Data Label ↓ | Class 0 | Class 1 | Undecided |
|---|---|---|---|
| Class 0 | 150 | 0 | 0 |
| Class 1 | 0 | 150 | 0 |

**Table D.2. A-HNN Confusion Matrix for XOR Data.**

The multi-layer perceptron and the A-HNN both provided a network that produced 100% testing accuracy. However the A-HNN generated a network with fewer number of nodes on the hidden layer, i.e., 5 nodes versus 10, and requiring less time to train than the generic multi-layer perceptron, i.e., 52.1M flops versus 78.6M, resulting in a substantial improvement in computer time and memory utilization. Training performance is substantially improved using the A-HNN concept of inputs as desired target values.

### *D.2 Material Data Classification*

The previous example extended the A-HNN for use with classification data. The XOR data set is an extremely simple classification problem and provides only the preliminary investigation for A-HNN effectiveness. Using XOR data, training was improved using one input as desired target output. The question of whether the additional target outputs must be duplicated features from the input, or simply additionally identified features (not input values at all) is still to be examined. (Due to the simplicity of the XOR data, this question could not be addressed.) The A-HNN effectiveness will be demonstrated using real-world material property data.

The materials data used for this experiments is ternary system material property data generated over many years of research. Typically, this data is separated into forming

and non-forming materials classifications. On-going materials research emphasizes the need to predict those three element compounds that will form new material from those that won't.

The initial step to this prediction is determining the features that effectively discriminate formers from non-formers. Some material properties are well documented in materials handbooks, primarily through experimental results, some of which are non-reproducible and non-verifiable. The A-HNN will be used for associating the experimental handbook data, i.e., chemical elemental properties, to a set of features representing the physical nature of the material, namely the atomic characteristics of each element [4]. This experiment will translate the experimentally determined features to physical property data that may provide more accuracy in classification of forming/non-forming materials.

For the first set of features, each of the three elements are identified by five variables (chemical elemental properties): melting point, atomic number, number of valence electrons, electronegativity, and the Zunger radii. The second set of features is the atomic characteristics of each element, the number of electrons in the five shells. The atomic characteristics of each element are non-experimental, therefore, the ability to associate these two feature sets will permit the usage of a more stable feature set than the experimentally gathered properties for determining formers and non-formers of yet to be defined compounds.

---

[4] Data courtesy of Air Force Research Laboratory, Materials Process Design Branch, AFRL/MLMR, 2977 P Street, Wright-Patterson AFB, OH 45433; Dr. Pierre Villars, Materials Phases Data System (MPDS), CH-6354 Vitznau, Switzerland; and Dr. Stephen Thaler, Imagination Engines, Inc., 12906 Autumn View Drive, St. Louis, MO 63146.

The A-HNN architecture, capable of predicting as well as classifying data, is applied to this effort using the calculated chemical elemental properties as inputs and the atomic characteristics of each element as the prediction target values as described in Chapter 3. The determination of the "autoassociative" target values requires multiple experiments, i.e., different program runs to define the optimum set of inputs to be used as target outputs. This research covers the evaluation of the A-HNN effectiveness under a variety of data conditions. A multi-layer perceptron is also programmed for a benchmark comparison.

The materials data set consists of a total of 4104 chemical elemental properties vectors, 15 features in length. 3500 of these vectors are used as training sample vectors. The prediction target values, the atomic characteristics of each element are determined for each of the 3500 three-element compounds. The prediction target values are also 15 features in length and consist of the number of electrons in the core, s-, p-, d-, and f-shells for each element. The input and prediction target vectors are graphically depicted in Figure D.1.

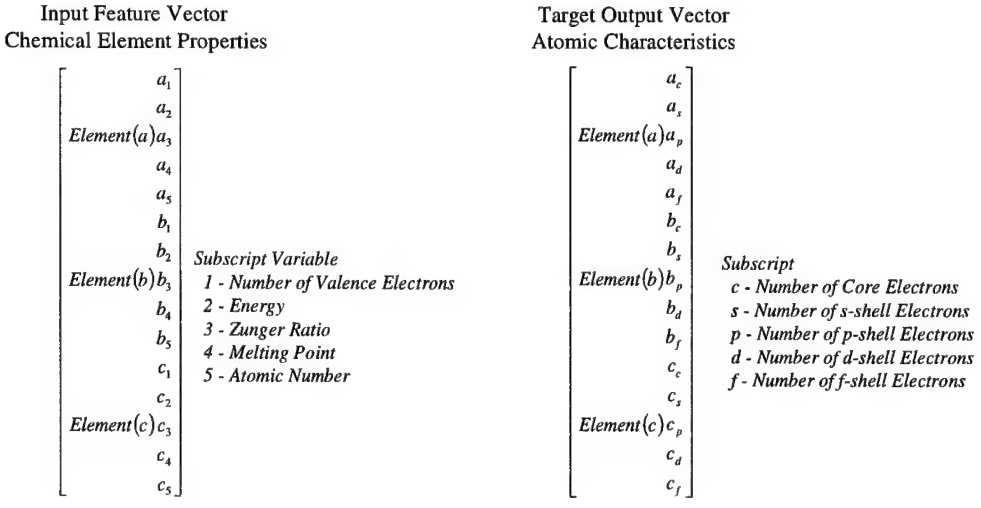



**Figure D.1. Input and Target-Output Materials-Data Vectors.**

Four distinct networks were programmed for this experiment. (The architecture for each of the trial runs was determined by trial and error network runs, offsetting the mean squared convergence of the weights with the testing accuracy as described in Appendix C.) The first network, a generic multi-layer perceptron, 15 feature inputs to 15 feature target values was implemented as a single-hidden-layer network with 15 nodes. This network trained the 15 feature-input vectors, Figure D.1, left, to the 15 feature target vectors. The second implementation, an A-HNN with the 15 feature inputs, and the target vector with the full autoassociative 15 feature inputs as the top half, and the desired 15 feature target values as the bottom half, was programmed with a single hidden layer with 25 nodes. The last two networks are A-HNNs programmed using variations of features on the input as well as for desired target values. One A-HNN variation uses all 15 features on the input, and a reduced subset of inputs as desired target values, through a single-hidden-layer network with 10 nodes. This input/target output configuration is depicted graphically in Figure D.2 (a). The second variation uses a reduced subset of the 15 features as inputs, with the target values from the 'rejected' input features. This configuration is graphically depicted in Figure D.2 (b).

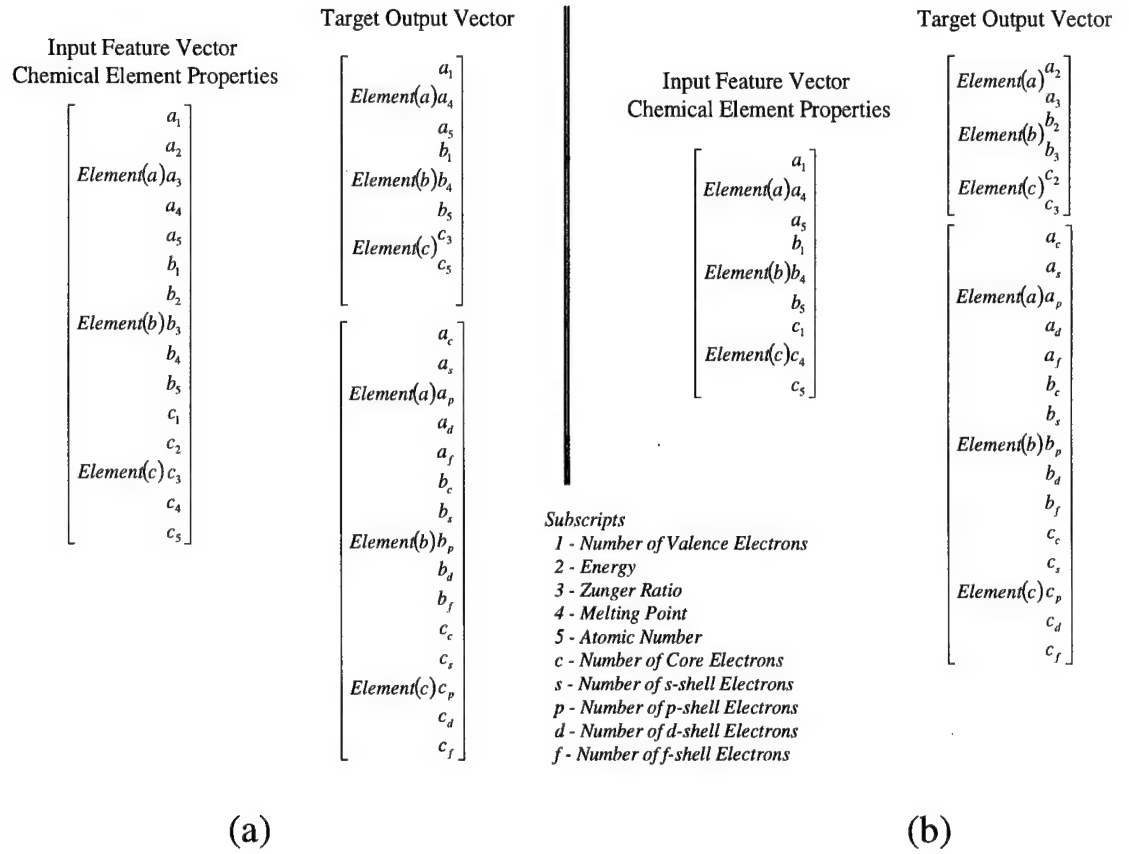



**Figure D.2. Input and Target-Output Configurations for Two Implementations of the A-HNN.**

These four networks were generated and run and the results are shown in Table D.3. The best overall accuracy is obtained using the A-HNN with all 15-feature inputs also used as target output values. For these results the network configuration required 25 nodes on the single hidden layer -- more than any other configuration, but it trained to a desirable testing accuracy in a fewer number of training epochs and only slightly more flops than a smaller configuration. The generic multi-layer perceptron provided the next best testing accuracy with a smaller network configuration and less number of flops. However, though the overall accuracy is over 92%, the individual feature testing accuracy is significantly lower than the individual feature testing accuracy

| Network Configuration | Epochs | Hidden Layer Nodes | Network Accuracy | Least Feature Accuracy | Number of Flops |
|---|---|---|---|---|---|
| A-HNN | 15,000 | 25 | 93.6% | 83.4% | 4.7955 E10 |
| Multi-Layer Perceptron | 35,000 | 15 | 92.2% | 68% | 4.3985 E10 |
| A-HNN: Subset of Inputs to Different Subset of Inputs as Targets | 30,000 | 20 | 88.8% | 73.8% | 5.3568 E10 |
| A-HNN: Full Set of Inputs to Subset of Inputs as Targets | 30,000 | 10 | 84.4% | 67.4% | 3.38619 E10 |

**Table D.3. Results of Network Implementations of Materials Data.**

in the full A-HNN configuration. The last two A-HNN configurations, with different features selected for the input and target output values, the results were significantly inferior, especially as regards the individual feature testing accuracy.

### *D.3 Conclusions*

For the two problems addressed here, the XOR classification and a real-world materials property data analysis, the A-HNN has been demonstrated as an effective and efficient tool to process this data. The A-HNN, while providing a robustness check on the network, improves the training performance over that of multi-layer perceptron implementations.

From investigations performed to date -- image processing, classification, or general analysis -- the A-HNN was most effective when some or all of the input-vector values are duplicated as target-outputs. The duplicity of the data, as a constraint device

on the encoding section, and a target on the decoding section, significantly improves the training performance and the overall efficiency of the network operation.

*Bibliography*


Baldi, Pierre, and Kurt Hornik. "Neural Networks and Principal Component Analysis: Learning from Examples Without Local Minima," Neural Networks, Vol. 2, No. 1, 1989, pp. 53-58.

Baron, Robert J. *The Cerebral Computer: An Introduction to the Computational Structure of the Human Brain.* Hillsdale, New Jersey: Lawrence Erlbaum, Associates, 1987, pp. 134-191.

Bloch, Isabelle. "Information Combination Operators for Data Fusion: A Comparative Review with Classification", IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans, Vol. 26, No. 1, January 1996, pp. 52-67.

Bracewell, R. N. *The Fourier Transform and Its Applications.* New York: McGraw-Hill, 1965.

Burt, Peter J., and Raymond J. Kolczynski. "Enhanced Image Capture Through Fusion", *1993 IEEE 4th International Conference on Computer Vision*, Vol. 4, 1993, pp. 173-182.

Chinzei, Kiyoyuki, Takeyoshi Dohi, Takashi Horiuchi, Yuji Ohta, Makoto Suzuki, Yasushi Yamauchi, Daijo Hashimoto, and Masakazu Tsuzuki. "Quantitative Integration Of Multimodality Medical Images", *Proceedings SPIE-International Society For Optical Engineering, Vol. 1808.* Bellingham, Washington: 1992, pp. 187-195.

Cottrell, Garrison W., Paul Munro; and David Zipser. "Image Compression by Backpropagation: A Demonstration of Extensional Programming", *Models of Cognition: A review of Cognitive Science*, Vol. 1, 1989.

Cybenko, George. "Approximation by Superposition of a Sigmoidal Function", Mathematics of Control, Signals, and Systems, Vol. 2, No. 4, 1989.

Cybenko, George. "Neural Networks in Computational Science and Engineering", IEEE Computational Science & Engineering, Spring 1996, Vol. 3, No. 1.

Daly, Scott. "The Visible Differences Predictor: An Algorithm for the Assessment of Image Fidelity". *Digital Images and Human Vision.* Cambridge, Massachusetts: MIT Press, 1993, pp. 179-206.

DeMers, David and Garrison Cottrell. "Non-Linear Dimensionality Reduction", University of California, San Diego, California, 1993.
Website: http//www-cse.ucsd.edu/users/gary/

Dusser de Barenne, H. G. and others. "Physiological Neuronography of the Cortico-Striatal Connections", Association for Research in Nervous and Mental Disease, Vol. 21, 1942, pp. 246-266.

Field, D.J. "Scale-Invariance and Self-Similar 'Wavelet' Transforms: and Analysis of Natural Scenes and Mammalian Visual Systems", *Wavelets, Fractals, and Fourier Transforms*, eds. M. Farge, J.C.R. Hunt, and J. C. Vassilicos, Oxford: Clarendon Press, 1990, pp. viii and 151-193.

Fischbach, Gerald D. "Mind and Brain". *Mind and Brain: Readings from Scientific American Magazine*. New York: W.H. Freeman and Company, 1993, pp. 1-14.

Gonzalez, Rafael C. and Richard E. Woods. *Digital Image Processing*. New York: Addison-Wesley Publishing Company, 1992, pp. 148-156, 504.

Graps, Amara. "An Introduction to Wavelets", IEEE Computational Science and Engineering, Vol. 2, No. 2, Summer 1995.

Grimson, W. E. L., G. J. Ettinger, S. J. White, T. Lozano-Perez, W. M. Wells, and R. Kikinis. "An Automatic Registration Method for Frameless Stereotaxy, Image Guided Surgery, and Enhanced Reality Visualization", IEEE Transactions on Medical Imaging, Vol. 15, No. 2, April 1996, pp. 129-140.

Grimson, W.E.L. "Medical Applications of Image Understanding", IEEE Expert, October 1995, pp. 18-28.

Hall, Charles F. and Hall, Ernest L. "A Nonlinear Model for the Spatial Characteristics of the Human Visual System", IEEE Transactions on Systems, Man, and Cybernetics, Vol. 7, No. 3, pp. 161-170.

Hecht-Nielsen, Robert. "Replicator Neural Networks for Universal Optimal Source Coding", Science, Vol. 269, 29 September 1995, pp. 1860 - 1863.

Hecht-Nielsen, Robert. *Neurocomputing*. New York: Addison-Wesley Publishing Co., 1990, pp. 325-330.

Heeger, David J. and Teo, Patrick C. "A Model of Perceptual Image Fidelity". *Proceedings of the 1995 IEEE International Conference on Image Processing*, pp. 343-345.

Hinton, Geoffrey E. "How Neural Networks Learn from Experience". *Mind and Brain: Readings from Scientific American Magazine*. New York: W.H. Freeman and Company, 1993, pp. 113-124.

Hubbard, Barbara Burke. *The World According to Wavelets*. Wesley, Massachusetts: A K Peters, Ltd., 1996, pp. 137-152, 187-202.

Kabrisky, Matthew. *A Proposed Model for Visual Information Processing in the Human Brain.* London: University of Illinois Press, 1966.

Kirby, M. and R. Miranda. "Empirical Dynamical System Reduction I: Global Nonlinear Transformations", Semi-Analytic Methods for the Navier-Stokes Equations, *Proceedings of the CRM Workshop*, Montreal, 1998.

Kirby, M. and R. Miranda. "Nonlinear Reduction of High-Dimensional Dynamical Systems via Neural Networks", Phys. Rev. Letters, Vol. 72, No. 12, 1994, pp. 1822.

Kramer, Mark A. "Nonlinear Principal Component Analysis Using Autoassociative Neural Networks", American Institute of Chemical Engineering Journal, Vol. 37, No.2, February 1991, pp. 233-243.

Krieg, W. J. S. *Functional Neuroanatomy.* New York: The Blakiston, Co., Inc., 1953.

Lin, Kang-Ping, Sung-Cheng Huang, Lewis R. Baxter, and Michael Phelps. "A General Technique for Interstudy Registration of Multifunction and Multimodality Images", IEEE Transactions on Nuclear Science, Vol. 41, No. 6, December 1994, pp. 2850-2855.

Linde, Y., A. Buzo and R. M. Gray. "An algorithm for Vector Quantizer Design", IEEE Transactions on Communications, Vol. COM-28, No. 1, Jan 1980.

Maher, Frank A. "A Correlation of Human and Machine Pattern Discrimination", *NAECON '70 Record*, pp. 260-264.

Mallat, Stephane G. "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 11, No. 7, July 1989.

Mallat, Stephane G. and Zhifeng Zhang. "Matching Pursuit with Time-Frequency Dictionaries", Technical Report 619, IEEE Transactions in Signal Processing, December 1993.

Malthouse, Edward C. "Some Theoretical Results on Nonlinear Principal Components Analysis", Ph.D. Dissertation. ftp website: ftp//mkt2715.kellogg.nwu.edu in pub/ecm/nlpca.ps, September 19, 1996.

Mansfield, P., and P. G. Morris. *NMR Imaging in Medicine.* Academic Press, Inc., 1982.

Martin, Curtis E. *Perceptual Fidelity for Digital Color Imagery.* Ph.D. Dissertation, Department of Electrical Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 1996.

McCulloch, W.S. "Why the Mind is in the Head". *The Hixon Symposium.* John Wiley and Sons, Inc., 1951.

McLachlan, Dan. "The Role of Optics in Applying Correlation Functions to Pattern Recognition", Journal of the Optical Society of America, Vol. 52, No. 4, April, 1962, pp. 454-459.

Murphy, Robin R. "Biological and Cognitive Foundations of Intelligent Sensor Fusion", IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, Vol. 26, No. 1, January 1996, pp. 42-51.

Nachmias, J. "On the Psychometric Function for Contrast Detection". *Vision Research* Vol. 21, pp. 215-223.

Newton, Thomas H. and D. Gordon Potts, Editors. *Radiology of the Skull and Brain, Volume 5, Technical Aspects of Computed Tomography.* The C. V. Mosby Company, Inc., 1981.

NOSTRA - Naval Ophthalmic Support and Training Activity: http://nos40.med.navy.mil/main.htm

Oja, Erkki. "Principal Components, Minor Components, and Linear Neural Networks", Neural Networks, Vol. 5, No. 6, 1992, pp. 927-935.

Oxley, Mark E. and Bruce Suter. "A Unified Theory of Feedforward Neural Networks". In Technical Review: Journal of Neural Networks, email: mark.oxley@afit.af.mil.

Padgett, Curtis, Garrison W Cottrell,. and Ralph Adolphs. "Categorical Perception in Facial Emotion Classification", 1998. Website: http//www-cse.ucsd.edu/users/gary/.

Pao, Yoh-Han. *Adaptive Pattern Recognition and Neural Networks.* Reading, Massachusetts: Addison-Wesley, 1989.

Pei, S.C. and M. H. Yeh. "An Introduction to Discrete Finite Frames", IEEE Signal Processing Magazine, November 1997, pp. 84-96.

Reimann, Stefan. "On the Design of Artificial Auto-Associative Neuronal Networks", Neural Networks, Vol. 11, 1998, pp. 611-621.

Rosenblatt. *Principles of Neurodynamics.* New York: Spartan Books, 1959.

Smith, Mark, J. T. and Thomas P. Barnwell III. "Exact Reconstruction Techniques for Tree-Structured Subband Codes", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP –34, No. 3, June 1986.

Soltanian-Zadeh, H., J. P. Windham and F. Chen. "Automated Contour Extraction Using A Multi-Scale Approach", *Proceedings IEEE Nuclear Science Symposium And Medical Imaging Conference*, Vol. 4, 1994, p.1797-1801.

Stockham Jr., Thomas G. "Image processing in the context of a visual model". Proceedings of the IEEE, Vol. 60, No. 7, pp. 828-842.

Tank, David W. and John J. Hopfield. "Collective Computation in Neuronlike Circuits". *The Workings of the Brain: Development, Memory, and Perception.* New York: W.H. Freeman and Company, 1990, 149-161.

Toet, Alexander, L.J. Van Ruyven, and J. M. Valeton. "Merging Thermal and Visual Images by a Contrast Pyramid", Optical Engineering, Vol. 28, No. 7, 1989, pp. 789-792.

Turk, Matthew and Alex Pentland. "Eigenfaces for Recognition", Journal of Cognitive Neuroscience, Volume 3, Number 1, Winter 1991, pp. 71-86.

Undrill, Peter E., G. G. Cameron, M. J. Cookson, Chris Davies, Neil L. Robinson, Andrew Hill, T. F. Cootes, Christopher J. Taylor, Ann Thornham, J. Wysocki, Heather M. Liddell, and Dennis Parkinson. "Integrated Presentation of 3-D Data Derived From Multisensor Imagery and Anatomical Atlases Using a Parallel Processing System", *Proceedings SPIE - The International Society For Optical Engineering*, Vol. 1653, 1992, p 2-16

Van den Elsen, P.A., J. B. A. Maintz, E. J. D. Pol, and M. A. Viergever. "Automatic Registration of CT and MR Brain Images Using Correlation of Geometrical Features", IEEE Transactions on Medical Imaging, Vol. 14, No. 2, June 1995, pp. 384-396.

Visible Human Project: National Institutes of Health, National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894; email: vhp@nlm.nih.gov.

Wang, Matthew Y., Calvin R. Maurer, Jr., Michael Fitzpatrick, and Robert J. Maciunas. "An Automatic Technique for Finding and Localizing Externally Attached Markers in CT and MR Volume Images of the Head", IEEE Transactions on Biomedical Engineering, Vol. 43, No. 6, June 1996, pp. 627-637.

Wasserman, Philip D. *Neural Computing, Theory and Practice.* New York: Van Nostrand Reinhold, 1989.

Westen, SJP and others. "Perceptual Image Quality Based on a Multiple Channel HVS Model". *Proceedings of the 1995 International Conference on Acoustics, Speech, and Signal Processing*, 1995, pp. 2351-2354.

Wilson, Terry Allen. "Perceptual Based Image Fusion with Applications to Hyperspectral Image Data", Master's Thesis, Department of Electrical Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 1995.

Zeki, Semir. "The Visual Image in Mind and Brain". *Mind and Brain: Readings from Scientific American Magazine*. New York: W.H. Freeman and Company, 1993, pp. 27-39.

Zeki, Semir. *A Vision of the Brain*. Oxford: Blackwell Scientific Publications, 1993a.

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | May 1999 | Ph.D. Dissertation |

**4. TITLE AND SUBTITLE**
Image Fusion Using Autoassociative-Heteroassociative Neural Networks

**5. FUNDING NUMBERS**

**6. AUTHOR(S)**
Claudia V. Kropas-Hughes

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433-6583

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
Air Force Research Laboratory/Materials and Manufacturing Directorate
AFRL/MLMR
2977 P Street, B653, R014
WPAFB, OH 45433
Dr. Steven LeClair, (937)255-8787

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
AFIT/DS/ENG/99-06

**11. SUPPLEMENTARY NOTES**

**12a. DISTRIBUTION AVAILABILITY STATEMENT**
Approved for public release; Distribution unlimited

**12b. DISTRIBUTION CODE**

**13. ABSTRACT** *(Maximum 200 words)*

Images are easily recognized, classified, and segmented--in short, analyzed--by humans. The human brain/nervous system--the biological "computer"--performs rapid and accurate image processing. In the current research the concepts of the biological neural system provide the impetus for developing a computational means of fusing image data. Accomplishing this automatic image processing requires features be extracted from each image data set, and the information content fused. Biologically- inspired computational models are examined for extracting features by transformations such as Fourier, Gabor, and wavelets, and for processing and fusing the information from multiple images through evaluation of autoassociative neural networks (AANNs). Features are obtainable through the use of human-visual-system models, however, AANNs are limited in accomplishing the desired data fusion. In-depth analysis of these networks demonstrated their functionality as data filters requiring careful feature selection to provide the desired image processing. Some features, when using perceptual space concepts, required AANNs to have the ability to process complex-valued inputs instead of only real-valued data. Human-visual-system concepts provided an alternative error function metric for training the networks based on the human standard of similarity instead of absolute numerical value. The most significant limitation of AANNs, for this data fusion application, was their inability to process multiple image data sources. The AANN concepts were extended to a new architecture--the Autoassociative-Heteroassociative Neural Network (A-HNN)--developed for predicting one sensor image from another by using input data values as desired targets.

**14. SUBJECT TERMS**
neural networks, image processing, human visual system, wavelets, autoassociative neural networks, heteroassociative neural networks, data filters, stability

**15. NUMBER OF PAGES**
132

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | SAR |